НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

«На правах рукопису»
УДК 004.043

До захисту допущено:
Завідувач кафедри
_______ Ігор ПАРХОМЕЙ
«___»_____2020 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-професійною програмою «Інформаційне забезпечення робототехнічних систем»

зі спеціальності 126 «Інформаційні системи та технології»

на тему: «Розробка системи розпізнавання тексту на зображенні»

Виконав:
студент ІІ курсу, групи ІК-91МП
Мальченко Євгеній Євгенійович _____

Керівник:
Старший викладач
Польшакова Ольга Михайлівна _____

Консультант з нормоконтролю:
доцент, к.т.н., доц.,
Пасько Віктор Петрович _____

Рецензент:
ст. викл. кафедри АУТС
Моргаль Олег Михайлович _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2020 року

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра технічної кібернетики

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційне забезпечення робототехнічних систем»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Ігор ПАРХОМЕЙ

«___»_____2020 р.

ЗАВДАННЯ

на магістерську дисертацію студенту

Мальченко Євгенію Євгенійовичу

1. Тема дисертації «Розробка системи розпізнавання тексту на зображенні», науковий керівник дисертації ст. викладач Польшакова О.М., затверджені наказом по університету від « 26 » жовтня 2020р. № 3132-с

2. Термін подання студентом дисертації 14.12.2020

3. Об'єкт дослідження – системи розпізнавання рукописних символів в умовах малої навчальної вибірки.

4. Предмет дослідження – недостатньо висока точність розпізнавання рукописних символів в системах, заснованих на алгоритмах, що працюють умовах малої навчальної вибірки.

5. Перелік завдань, які потрібно розробити – Проаналізувати існуючі системи розпізнавання тексту на зображенні, виділити їх недоліки і переваги, проаналізувати існуючі алгоритми розпізнавання тексту на зображенні, проаналізувати існуючі алгоритми попередньої обробки зображення, розробити покращений алгоритм розпізнавання рукописних символів, розробити програмне забезпечення для системи, що включає реалізацію попередньої обробки зображення та розпізнавання тексту на зображенні, перевірити ефективність роботи запропонованої системи.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу – 6 плакатів

7. Орієнтовний перелік публікацій – одна публікація

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Перевірка на співпадіння	Лісовиченко О.І., доцент	03.12.20	03.12.20
Нормоконтроль	Пасько В.П., доцент	10.12.20	10.12.20

9. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз існуючих систем розпізнавання тексту на зображенні	01.10.2020-04.10.2020	
2	Аналіз існуючих алгоритмів розпізнавання тексту на зображенні	05.10.2020-09.10.2020	
3	Аналіз існуючих алгоритмів попередньої оброки зображення	10.10.2020-11.10.2020	
4	Розробка покращеного алгоритму розпізнавання рукописних символів	12.10.2020-18.10.2020	
5	Розробка програмного забезпечення для системи, що включає реалізацію попередньої обробки зображення та розпізнавання тексту на зображенні	19.10.2020-08.11.2020	
8	Перевірка ефективності розробленої системи	09.11.2020-13.11.2020	
9	Оформлення пояснювальної записки	14.11.2020-30.11.2020	
10	Попередній захист	23.11.2020	
11	Нормоконтроль	03.12.2020	
12	Перевірка на співпадіння	10.12.2020	
13	Захист	22.12.2020	

Студент Євгеній Мальченко

Науковий керівник Ольга Польшакова

АНОТАЦІЯ

У представленій роботі розглянуто розробку системи розпізнавання тексту на зображенні.

Було проведено аналіз існуючих систем та методів розпізнавання тексту на зображенні, виявлено їх основні недоліки і переваги. Проаналізовано алгоритми розпізнавання та попередньої обробки зображення. Запропоновано власний продуктивний алгоритм попередньої обробки зображення, алгоритм побудови структурної моделі символу, та критерій схожості побудованих структурних моделей символів. Розроблено програмну реалізацію системи розпізнавання тексту на зображенні, здатну працювати з високою точністю при невеликій еталонній вибірці. Виконано перевірку ефективності розробленої системи та порівняння із аналогами.

Результатом виконаної роботи є розроблена система розпізнавання тексту на зображенні, яка виконує поставлені перед нею задачі з високою точністю та продуктивністю.

Ключові слова: Розпізнавання тексту, обробка зображення, бінаризація, скелетизація, структурна модель, критерій схожості.

Розмір пояснювальної записки – 111 сторінок, 53 ілюстрації, 33 таблиці, 3 додатки.

ABSTRACT

This paper considers the development of a text recognition system in the image.

The analysis of existing systems and methods of text recognition in the image was carried out, their main disadvantages and advantages were revealed. Algorithms for image recognition and pre-processing are analyzed. The own productive algorithm of image pre-processing, algorithm of construction of structural model of symbol, and criterion of similarity of constructed structural models of symbols are offered. A software implementation of a text recognition system in an image capable of working with high accuracy with a small reference sample has been developed. The efficiency of the developed system was checked and compared with analogues.

The result of the work performed is a developed text on the image recognition system that performs the tasks set before it with high accuracy and productivity.

Keywords: Text recognition, image processing, binarization, skeletization, structural model, similarity criterion.

The size of the explanatory note is 111 pages, 53 illustrations, 33 tables, 3 appendices.

Пояснювальна записка
до магістерської дисертації

на тему: ***Розробка системи розпізнавання тексту на зображенні***

Київ – 2020 року

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

OCR – Оптичне розпізнавання символів

SMM – Алгоритм на основі максимального паросполучення мінімальної ваги

SVM – Алгоритм на основі структурних векторів

ІНМ – Імовірнісна нейронна мережа

ШНМ – Штучна нейронна мережа

ВСТУП

На сьогоднішній день системи розпізнавання символів забезпечують вирішення низки наукових і прикладних задач, що виникають в процесі вилучення текстової інформації з друкованих та рукописних документів [1, 2]. Розроблено безліч методів розпізнавання символів [3, 4, 5], серед яких: методи на основі ознакових класифікаторів (штучні нейронні мережі, метод опорних векторів, карти Кохонена та ін.); статистичні методи (підходи з побудовою гістограм, метод перетинів, методи на основі зонного опису та ін.); методи на основі виділення структурних складових, які засновані на виділенні певних геометричних властивостей зображення символу і подальшій побудові моделі, яка описує символ із застосуванням виділених геометричних властивостей [6, 7].

Всі існуючі універсальні підходи до вирішення задачі розпізнавання символів орієнтовані на застосування опорної бази еталонних зображень, яка для високої точності розпізнавання повинна бути досить велика. У той же час існує ряд завдань, в яких кількість спочатку відомих накреслень символів вкрай невелика. Прикладами таких завдань є: розпізнавання заповнених нетиповим почерком бланків атестації, виділення текстової інформації на наявних в єдиному екземплярі історичних документах, ідентифікація підписів в банківських документах, ідентифікація користувача по рукописного підпису, тощо.

Особливу складність в умовах малої кількості еталонних зображень представляє завдання сегментації рукописного тексту. В даний час для її рішення застосовуються ознакові класифікатори, але в умовах малої еталонної вибірки виконати якісне навчання таких класифікаторів не представляється можливим через те, що замість необхідних для навчання лігатур (пар суміжних символів) серед еталонних зображень є тільки розрізнені символи. Загальновідомі методи без застосування ознакових класифікаторів спираються на використання вертикальних ліній для поділу сегментів в слові, тим самим обмежуючи можливість коректної сегментації тексту, написаного почерком з великим нахилом.

У зв'язку з цим проблема розробки системи розпізнавання тексту на зображенні, заснованої на побудові структурних моделей і здатних функціонувати в умовах малої вибірки еталонних зображень, є актуальною.

Об'єктом дослідження є системи розпізнавання рукописних символів в умовах малої навчальної вибірки.

Предмет дослідження: недостатньо висока точність розпізнавання рукописних символів в системах розпізнавання тексту на зображенні, заснованих на алгоритмах, що працюють в умовах малої навчальної вибірки.

Метою дисертаційної роботи є розробка системи розпізнавання тексту на зображенні в умовах малої навчальної вибірки.

Наукову новизну роботи представляє внесок у розвиток систем розпізнавання тексту на зображенні, що полягає в розробці системи розпізнавання рукописних символів в умовах малої навчальної вибірки із використанням запропонованого багатоетапного алгоритму розпізнавання рукописних символів, що включає стадії попередньої обробки зображення, виділення структурних складових чотирьох типів, побудови структурної моделі символу на основі цих складових, а також критерії схожості структурних моделей символів. Запропоновані методи мають високу швидкодію і забезпечують можливість розпізнавання рукописних символів з різними особливостями накреслення. Головною особливістю розробленої системи розпізнавання тексту на зображенні є можливість якісної класифікації при малій кількості еталонних зображень символів, якої недостатньо для повноцінного навчання ознакових класифікаторів.

РОЗДІЛ 1. ЗАГАЛЬНИЙ ОГЛЯД ІСНУЮЧИХ РІШЕНЬ ТА АНАЛІЗ МЕТОДІВ І АЛГОРИТМІВ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННІ

У цьому розділі представлено аналітичний огляд систем, методів і алгоритмів, що застосовуються для розпізнавання символів. Проаналізовано їх переваги та недоліки, сформовано детальні порівняльні таблиці оглянутих методів та заснованих на них систем, на підставі чого зроблено вибір на користь групи методів на основі структурних складових.

1.1 Основні особливості задачі розпізнавання символів

Завдання розпізнавання символів трохи простіше завдання класифікації довільних образів на зображеннях. Як правило, рішення задачі розпізнавання символів не має на увазі необхідність відділення кордонів графічного представлення символу від фону. Якщо вважати, що кожен з пікселів аналізованого зображення належить або до графічного представлення символу, або до простого фону, то кількість можливих подань символу істотно менше, ніж кількість можливих подань реального об'єкта, наприклад, на фотознімках. Однак через те, що завдання розпізнавання має ряд істотних спрощень по відношенню до завдань розпізнавання більш складних об'єктів, вимоги до алгоритмів і методів вирішення цього завдання істотно вище.

1.2 Відмінності завдань розпізнавання друкованих та рукописних символів

На відміну від друкованих символів, рукописні символи мають набагато більшу кількість графічних уявлень. Один і той же символ може бути написаний різними людьми, кожен з яких має власний почерк. Крім того, можуть варіюватися матеріал, на якому виконується накреслення, і інструмент, яким воно виконується.

Людський мозок здатний відрізняти рукописні символи з деяких особливостей їх графічного представлення і апріорної інформації про можливі способи накреслення певних видів символів. Така інформація накопичується в пам'яті людини протягом усього його життя, в той час як автоматизована система

набагато більш обмежена в обсягах використовуваної пам'яті і часу, який потрібен їй для налаштування або навчання.

1.3. Існуючі методи розпізнавання символів

Існуючі підходи можна розділити на наступні категорії:

- методи з використанням ознакових класифікаторів;
- статистичні методи;
- методи з використанням структурних складових.

Перша група методів є однією з найбільш широко поширених в науковому середовищі. При розробці таких методів можна абстрагуватися від деталей процесу порівняння образів, поклавши цю необхідність на один з математичних інструментів для класифікації. Досить широкого поширення набули в задачах розпізнавання не тільки символів, але і образів в цілому штучні нейронні мережі. Останнім часом для подібних завдань використовуються машини опорних векторів. На сьогодні у всесвітній мережі Інтернет у вільному доступі існує величезна кількість бібліотек, що реалізують ті чи інші ознакові класифікатори. З огляду на це, процес реалізації стає порівняно нетрудомістким, а рішення задачі розпізнавання зводиться до правильного вибору простору ознак і репрезентативною навчальної та тестової вибірки образів. Величезним недоліком цієї групи методів є відсутність можливості надати зручний для сприйняття людиною аргументовану звіт про те, чому образ віднесений саме до того чи іншого класу.

Друга група методів так само, як і описана раніше, досить проста в плані програмної реалізації. Початкове графічне представлення образу аналізується з метою виявлення різних статистичних величин, будуються гістограми, вважаються середні значення і середньоквадратичні відхилення в певних областях. Після чого отримані дані використовуються для виконання класифікації. Для класифікації може використовуватися, як простий ознаковий класифікатор, так і евристичний підхід з використанням виявлених закономірностей. Гістограми набагато більш наочні для людського сприйняття і можуть бути корисні в процесі налагодження або поліпшення алгоритму.

Третя група методів відрізняється тим, що пошук закономірностей і залежностей графічного вигляду символів покладається на розробника методу. На графічному поданні символу виділяються ключові елементи – структурні складові. Вони можуть бути представлені, наприклад, графічними примітивами або послідовністю дій оператора при зображенні символу. Витягти з графічного накреслення подібні складові – досить трудомістке завдання, але ще більш важкою є задача їх використання для порівняння двох символів між собою. Як розробка, так і програмна реалізація таких методів займає, як правило, значну кількість часу. Однак, подібні алгоритми, мають більш високу точність розпізнавання в завданнях, де є одна або декілька еталонних форм графічного представлення символу. При розробці алгоритмів цієї групи доводиться вирішувати задачу знаходження «золотої середини» між досить точним поданням розбиття на структурні складові і не дуже складним поданням цього розбиття. Адже в такому випадку процес порівняння може виявитися надто вимогливим до обчислювальних потужностей.

1.3.1 Методи розпізнавання символів з використанням ознакових класифікаторів

Штучні нейронні мережі (ШНМ) [9] знайшли широке застосування в задачах розпізнавання символів. ШНМ – математична модель, що представляє із себе множина штучних нейронів, з'єднаних між собою за допомогою синаптичних зв'язків. Нейрони згруповані по верствам. Як правило, виділяють вхідний, вихідний і приховані шари. На вхід ШНМ надходить вектор ознак, що описує певний образ. Розмірність простору ознак дорівнює кількості нейронів у вхідному шарі. Вихідний шар містить кількість нейронів, необхідне для зберігання інформації про результат класифікації [10]. Найчастіше для кожного можливого класу у вихідному шарі виділяється окремий нейрон. Навчанням ШНМ називається етап настройки ваг її синаптичних зв'язків. Стадія навчання, фактично, є стадією побудови простору ознак класифікатора на основі ШНМ.

Вхідний вектор ознак для класифікатора на основі ШНМ часто виходить з використанням вейвлет-перетворення, яке є згорткою вейвлет-функції з

вихідним сигналом [10, 11]. У статті наукової групи з Індії в складі V.N. Manjunath Aradhya, S.K. Niranjan і G. Hemantha Kumar в 2010 році запропоновано використання ймовірнісної нейронної мережі для розпізнавання рукописних символів. Імовірнісна нейронна мережа або, скорочено, PNN-мережа – одна з різновидів радіально-базисних мереж.

Імовірнісна нейронна мережа має три шари: вхідний, радіальний і вихідний. Вхідний шар має довільну розмірність. Кожен нейрон радіального шару відповідає одному елементу навчальної вибірки. Кількість нейронів вихідного шару дорівнює кількості класів. Кожен нейрон вихідного шару з'єднаний з елементами радіального шару, що належать відповідним йому класу (рис. 1.1.).

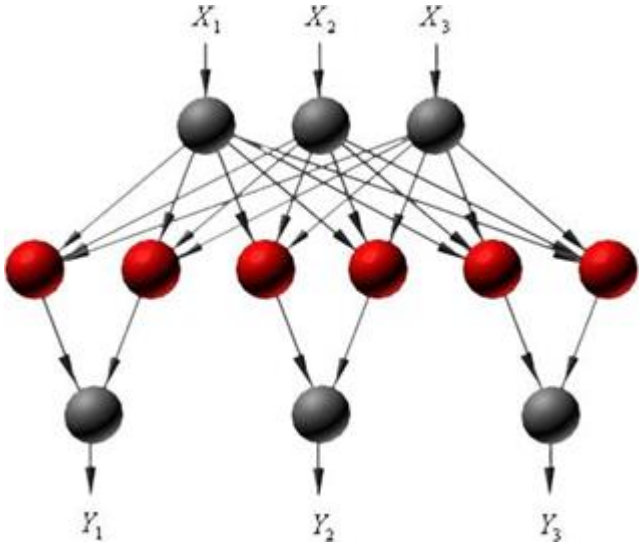


Рисунок 1.1. Структура імовірнісної нейронної мережі

За своєю суттю принцип роботи PNN-мережі [12] ґрунтується на так званих ядерних оцінках. Для оцінки щільності ймовірності приналежності вектора ознак певного класу використовується інформація про наявність в безпосередній близькості до цього вектору ознак векторів, відповідних цього класу в радіальному шарі ймовірнісної нейронної мережі.

Для отримання вектора ознак графічних зображень символів використовується дискретне перетворення Фур'є. Після цього отриманий вектор подається на вхідний шар ймовірнісної нейронної мережі. Нейрон вихідного шару з найбільшим значенням радіальної функцій відповідає класу, до якого слід віднести зображення [13, 14, 15].

До істотних переваг такого алгоритму можна віднести імовірнісний сенс вихідних значень штучної нейронної мережі. Такі значення згодом можуть бути використані для поліпшення якості класифікації з урахуванням певної апріорної інформації, наприклад, словника. Результати вчених з Індії для букв англійського алфавіту сильно залежать від кількості зображень в навчальній вибірці. Якщо використовувати 175 прикладів зображень кожного класу, то відсоток правильно розпізнаних символів знаходиться в інтервалі від 90% до 95%.

Таку сильну залежність результатів розпізнавання від кількості елементів в навчальній вибірці можна пояснити найбільш високою якістю апроксимації щільності ймовірності при більш високому числі елементів в навчальній вибірці. Варто також відзначити, що швидкодія PNN-мережі лінійно залежить від розміру радіального шару, і, відповідно, від розміру навчального набору зображень. Таким чином, за більш високу якість розпізнавання доводиться «розплачуватися» менш високою швидкодією.

Такий високий відсоток розпізнавання рукописних символів можна зв'язати і з тим, що велика частина елементів навчальної та тестової вибірки була виконана одним почерком без зайвих деформацій накреслення.

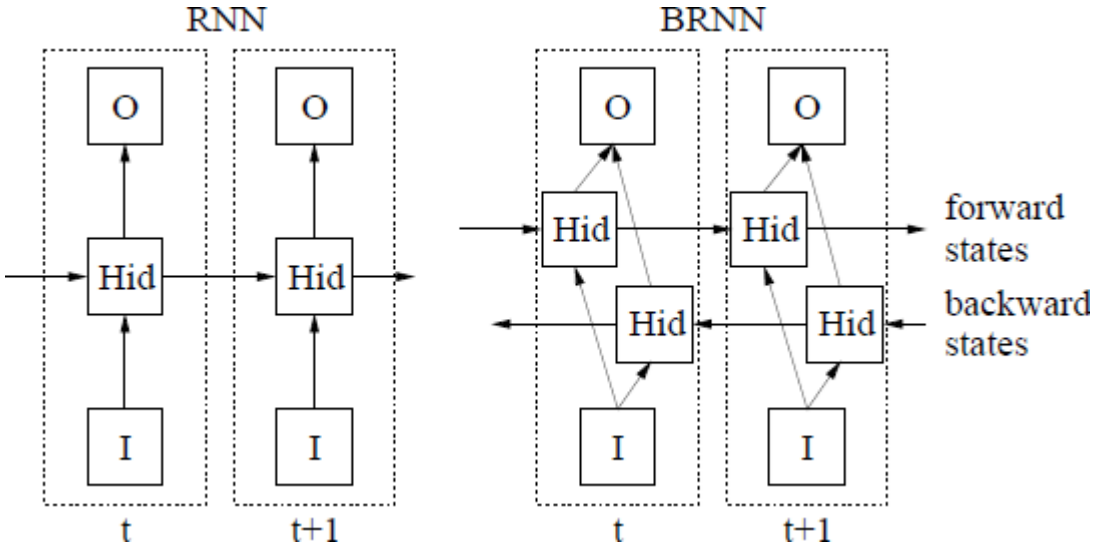


Рисунок 1.2. Рекурентна ІНС на основі архітектури Long Short-Term Memory і її аналог, розгорнутий у часі

Групою вчених з Берна в складі Marcus Liwicki, Alex Graves, Horst Bunke і Jurgen Schmidhuber в 2010 році був запропонований підхід для здійснення

сегментації рукописного тексту з використанням рекурентної штучної нейронної мережі на основі архітектури Long Short-Term Memory. Особливість цієї моделі полягає в тому, що деякі з вузлів штучної нейронної мережі здатні зберігати інформацію про останніх значеннях функції активації, які були отримані для попередніх моментів часу.

Величезною перевагою цієї моделі є можливість online-сегментації тексту, що вводиться [16].

Для навчання такого роду ШНМ використовуються зображення заздалегідь сегментированого тексту. За відгуками рекурентної нейронної мережі можна визначити, чи є певна позиція зображення з'єднанням двох рукописних символів (рис. 1.3).

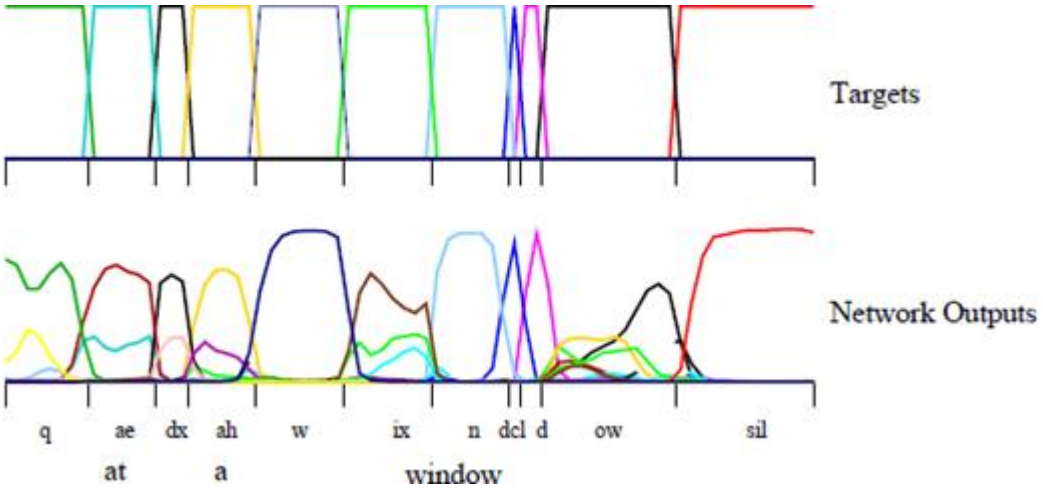


Рисунок 1.3. Приклад візуалізації використання відгуків ІНС

До того, як запропонувати такий підхід, автори використовували приховану марковскую модель, що дозволяло правильно сегментувати рукописний текст в 65,4% випадків. Використання ж нового підходу призводить до поліпшення даного показника до 74%. В майбутньому автори планують використовувати статистичні властивості мови для поліпшення якості сегментації, і, в результаті, розпізнавання рукописних символів.

Ще одна біологічно-подібна мережа, заснована на двох принципах роботи мозку: ієрархічного представлення об'єктів і використання тимчасової складової в процесі зору – ієрархічна тимчасова мережа, розроблена в ТПУ Ю.А. Болотової, використовується для розпізнавання рукописних і друкованих символів, в тому числі і для розпізнавання автомобільних номерів [17].

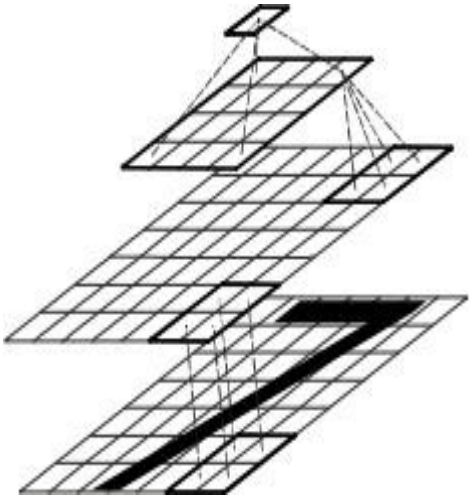


Рисунок 1.4. структура штучної часової нейронної мережі

Така мережа представлена кількома рівнями. Кожен з рівнів є двовимірною решіткою вузлів. Вузли верхніх шарів пов'язані з вузлами нижніх своїми рецептивними полями, які охоплюють кілька сусідніх вузлів нижнього шару.

Також група вчених з Берна запропонувала використовувати метод класифікації на основі трьох різних прихованих марковських моделей для вирішення задачі сегментації рукописного тексту. Марковська модель представляє роботу процесу, схожого з марковським процесом з невідомими параметрами. Знайдені параметри можуть використовуватися для подальшого аналізу, наприклад, для вирішення задачі класифікації зображень.

Російські вчені В. Вапник і А. Червоненкис запропонували метод опорних векторів – набір алгоритмів для вирішення задач класифікації та регресійного аналізу [18].

Метод опорних векторів нерідко використовується для розпізнавання друкованих і рукописних символів. Застосування методу в такому випадку зводиться до отримання деякого багатовимірного вектора ознак графічного представлення символу. Вектора ознак формують в просторі ознак групи, які потрібно розділити гіперплощиною так, щоб відстань до найближчої з площин було максимальним. Зі збільшенням відстані від гіперплощини до найближчого класу буде збільшуватися і точність класифікації. Вектори, розташовані найближче до шуканої гіперплощини називаються опорними векторами.

У разі якщо вибраний простір ознак лінійно нероздільний, потрібно відобразити цей простір в простір більшої розмірності так, щоб умова лінійної

роздільності виконувалося. Однак варто зазначити, що з ростом розмірності простору збільшується і складність процесу знаходження оптимальної гіперплощини.

Метод знайшов широке застосування в задачах розпізнавання символів, володіючи особливою перевагою в задачах, де навчальна вибірка має порівняно невеликі розміри. Істотним недоліком даного методу є той факт, що найбільший вплив на процес класифікації надають вектора, які знаходяться на кордонах множин.

Ще в 1998 році Y. LeCun спільно з L. Bottou, Y. Bengio і P. Haffner розробив згорткову нейронну мережу LeNet-5, призначену для розпізнавання рукописних цифр [19]. Архітектура представленої мережі приведена на рис. 1.5.

Тестування запропонованого класифікатора виконувалося на бенчмарковому наборі рукописних цифр MNIST, застосування якого передбачає навчання з використанням 60000 зображень, а тестування – на 10000 зображеннях аналогічного типу. Помилка навченої згорткової нейронної мережі склала лише 0,95% на цьому наборі.

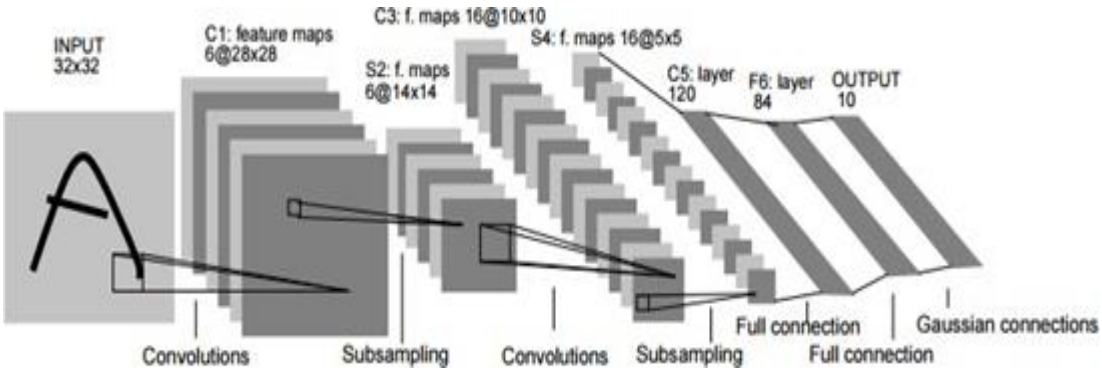


Рисунок 1.5. Архітектура LeNet-5 для розпізнавання рукописних цифр

У 2006 році D. Bouchain досліджував застосування мережі LeNet-5, а також попередніх її версій, в складі ансамблів класифікаторів все на тому ж загальновідомому наборі рукописних цифр MNIST. В результаті застосування ансамблю з різних версій мереж архітектури LeNet була досягнута помилка, яка дорівнює 0,7%.

Сьогодні згорткові нейронні мережі застосовуються для комплексного вирішення завдання виявлення і розпізнавання текстів на складному тлі.

1.3.2 Статистичні методи розпізнавання символів

Вчені E. Kavallieratou, N. Fakotakis і G. Kokkinakis з грецького міста Патри в 2008 році запропонували для розпізнавання окремих рукописних символів використовувати підхід, заснований на аналізі різних видів гістограм. Крім досить поширених вертикальної і горизонтальної гістограм запропоновано використовувати радіальну гистограмму і дві характеристики, які отримали назви «in-out профіль» і «out-in профіль» [20, 21].

Суть радіальної гістограми полягає в отриманні даних не для рядків або стовпців, а для прямих, проведених через центр під різними кутами.

In-out профіль будується шляхом знаходження найбільш близьких до центру пікселів, що належать символу, для кожної з аналогічних прямих. Точно так же будується і out-in профіль, якщо замість найбільш близького до центру пікселя знаходити найбільш віддалений.

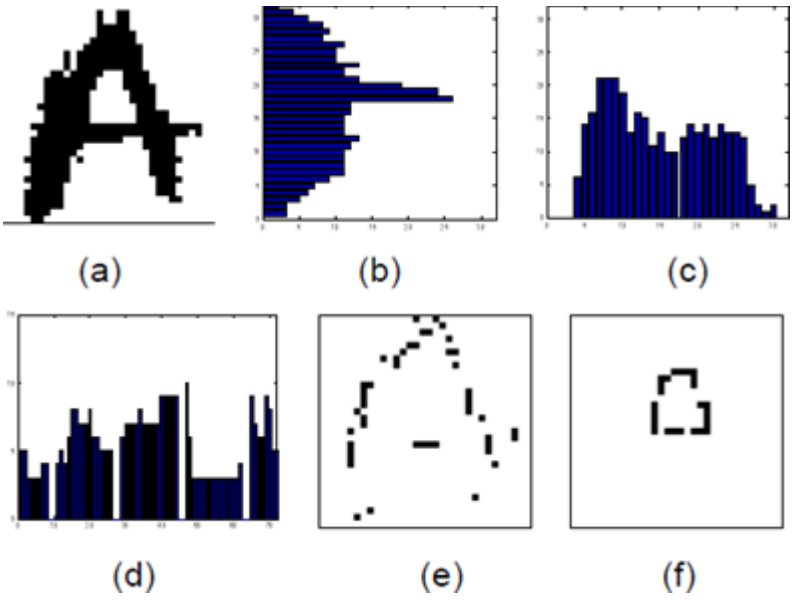


Рисунок 1.6. (a) зображення символу, (b) горизонтальна гістограма, (c) вертикальна гістограма, (d) радіальна гістограма, (e) out-in профіль, (f) in-out профіль

Незважаючи на досить простий спосіб отримання вектора ознак, автори методу опублікували результати оцінки якості розпізнавання, які показують, що подібний метод дуже надійний і з високим ступенем достовірності може розпізнавати рукописні цифри і букви латинського алфавіту.

Для оцінки якості розпізнавання використовувалося два відомих набору даних: NIST – для англійської мови, і GRUHD – для грецької мови. У кожному з цих наборів було зроблено три вибірки для оцінки серед цифр, малих літер, вживання великих літер та загального набору.

Дані про результати оцінки якості розпізнавання для набору NIST наведені в таблиці 1.1. Варто відзначити, що на одній з вибірок цифр цього набору запропонований алгоритм досяг стовідсоткової якості розпізнавання.

Для набору символів GRUHD результати дещо нижче, що цілком узгоджується з більш високою складністю набору і менш поширеною грецькою мовою. Проте, результати роботи алгоритму як і раніше залишаються високими (таб. 1.2):

Таблиця 1.1

Апробація алгоритму на наборі даних NIST

	Выборка №1	Выборка №2	Выборка №3
Цифри	98,80%	99.91%	100%
Малі букви	93,85%	96,54%	98,86%
Великі букви	91,40%	94,50%	98,85%
Змішана вибірка	82,79%	89,27%	96,85%

Таблиця 1.2

Апробація алгоритму на наборі даних GRUHD

	Выборка №1	Выборка №2	Выборка №3
Цифри	94,00%	97.42%	99,54%
Малі букви	86,03%	96,54%	98,96%
Великі букви	81,00%	90,36%	96,60%
Змішана вибірка	72,80%	80,04%	88,83%

В обох випадках істотне поліпшення якості розпізнавання на більш пізніх вибірках пояснюється тим, що попередні вибірки включаються в якості зразків для навчання при аналізі чергової вибірки [22].

До статистичних підходів можна віднести і групу методів на основі методу перетинів. Суть методу перетинів полягає в використанні деякого набору прямих, кожна з яких накладається на графічне представлення символу. Для кожної прямої підраховується кількість відрізків, які вона перетинає. З певних кількостей перетинів і формується вектор ознак, який використовується для подальшого виконання класифікації.

Перевагою цієї групи методів є їх інваріантність до дисторсії і невелика стилістичною варіація символів [23], а також досить висока швидкодія і низьке споживання оперативної пам'яті. Існують реалізації алгоритмів на основі методу перетинів без використання операцій над числами з плаваючою точкою.

Для вибору множини точок, що використовується в таких методах, використовують, як різні евристики та спостереження, так і еволюційні алгоритми. Як правило, вибір оптимальної множини для методу перетинів залежить від типу символів, з якими матиме справу алгоритм, що розробляється.

1.3.3 Методи розпізнавання символів на основі виділення структурних складових

Вчені університету The Hong Kong University of Science and Technology Kam-Fai Chan і Dit-Yan Yeung в 1999 році розробили більш зрозумілий для людського сприйняття підхід [24]. Ними було запропоновано використовувати аналіз на основі гнучкого порівняння структури. Основною відмінною здатністю є спосіб отримання дескрипторів. Дескриптори в цьому методі не будуть являти собою якісь абстрактні числові значення, які є деякими частотними характеристиками або усередненими значеннями рівнів яскравості будь-яких частин зображення. Автори пропонують аналізувати графічне представлення символу. Для початку розглядається мова представлення зображень на основі PDL-виразів з використанням чотирьох бінарних операцій:

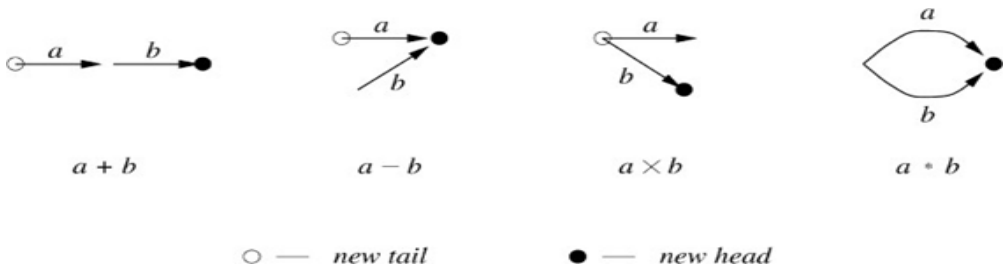


Рисунок 1.7. Типи бінарних операцій для задання PDL-виразів

Якщо додати до цих чотирьох операцій ще одну унарну операцію заперечення, то можна побудувати деякий набір накреслень:

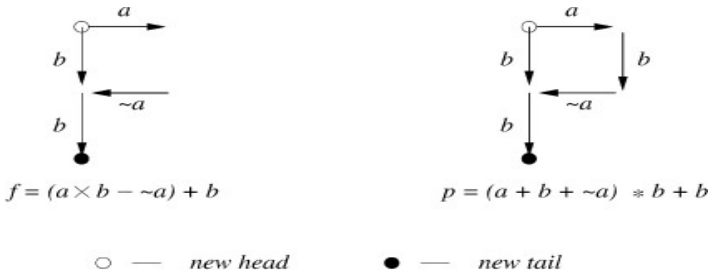


Рисунок 1.8. Приклади візуалізації PDL-виразів

Як можна помітити, такі вирази дозволяють задати лише невелику частину геометричних уявлень деяких символів. Деякі символи і зовсім можуть бути задані множиною різних виразів.

Інший спосіб опису структури символів – схема Бертода і Маро (Berthod and Maroy's encoding scheme). Ця схема використовує для опису геометричного представлення п'ять примітивів:

- Пряма лінія – T;
- Дуга за годинниковою стрілкою – P;
- Дуга проти годинникової стрілки – M;
- Перенесення пера – L;
- Виступ – R.

Комбінуючи такі примітиви можна кодувати кожне зображення символу деякої рядком:

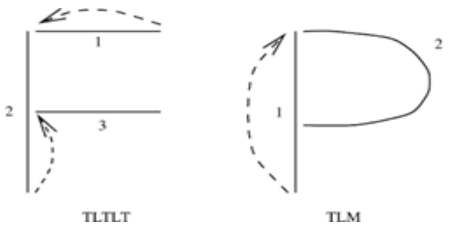


Рисунок 1.9. Приклади об’єктів в схемі Бертода і Марою

Як і у всіх попередніх моделей представлення, у цій моделі залишається можливість завдання одного і того ж символу двома або більше способами. Однак, кількість способів задати один і той же символ набагато менше, ніж у всіх раніше розглянутих способів.

Ще однією схемою, яку ми розглядаємо, є ланцюговий код Фрімана [26]. Код використовує вісім значень від 0 до 7, які задають, в якому напрямку з'єднуються два вузла:

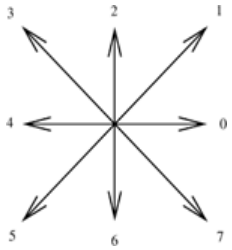


Рисунок 1.10. Напрямки в коді Фрімана

Спираючись на код Фрімана, можна розробити власну граматику, яка оперує з наступними примітивами:

- Відрізок – деяка послідовність пікселів, що належать графічному представленню символу, який візуально представляє собою пряму лінію;
- Дуга за годинниковою стрілкою – деяка послідовність пікселів, що належать до графічного представлення символу, яка візуально розташована істотно вище прямої лінії, що з'єднує два кінці цієї послідовності;
- Дуга проти годинникової стрілки – деяка послідовність пікселів, що належать до графічного представлення символу, яка візуально розташована суттєво нижче прямої лінії, що з'єднує два кінці цієї послідовності;
- Петля – деяка послідовність пікселів, що належать графічного представлення символу, яка починається і закінчується в одному і тому ж пікселі;
- Точка – деяка послідовність пікселів, що належать до графічного представлення символу, що складається з невеликої кількості пікселів. Найчастіше подібні області є піксельним шумом, але відкидати такі

області заздалегідь неприпустимо, адже вони можуть бути складовою частина таких символів, як «і» або «j».

Далі для зберігання інформації про структуру символу використовується більш інформативна граматика, ніж в кожному з раніше описаних способів.

Для визначення типу символу по його структурі використовується база зразків. Для кожного із зразків цієї бази заздалегідь визначається його структура, а потім черговий символ порівнюється з кожним із символів бази. Очевидно, що деякі символи мають кілька видів накреслення через особливості почерку або стилю написання. Для обліку подібних випадків при порівнянні виконуються деякі перетворення структури символів з бази, після чого проводиться тривіальне порівняння. За базу зображень рукописних символів була використана база «MIT Spoken Language Systems», яка містить графічні представлення рукописних накреслень 62 класів символів – цифри (0 – 9), малі літери англійського алфавіту (a – z), великі літери англійського алфавіту (A – Z).

Символ кожного класу в цій базі був накреслений одним з 150 людей власним почерком. Нижче наведені приклади накреслень символу «3» з цієї бази:

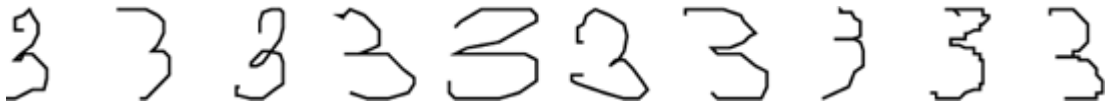


Рисунок 1.11. Приклади зображень «MIT Spoken Language Systems»

Як можна помітити, деякі символи були спеціально введені неакуратно, деякі – містять шуми. В цілому ж, вибірка містить лише зображення цифр, які виконувалися з урахуванням єдиного шаблону написання.

Результати оцінки якості розпізнавання для підготовленої вибірки наведені в таблиці 1.3.

Таблиця 1.3

Якість розпізнавання запропонованого алгоритму на підготованій вибірці

Цифри	98,60%
Малі букви	98,49%
Великі букви	97,44%
Загальний набор	97,40%

За результатами можна зробити висновок про те, що запропонований метод дозволяє з високою точністю визначати клас зображення рукописного символу без використання будь-яких ознакових класифікаторів: штучних нейронний мереж, радіально-базисних мереж, що самоорганізуються карт Кохонена, машин опорних векторів і т.д.

У зарубіжній літературі можна зустріти множина способів розпізнавання і сегментації рукописного тексту з використанням частотних характеристик і подальшого застосування інструментів нечіткої логіки або штучних нейронних мереж. Більшість цих методів мають точність розпізнавання, що перевищує 90%, проте точні критерії, за якими той чи інший символ був віднесений до певного класу, залишаються невідомими через відсутність чітко сформульованої логіки прийняття рішень.

На загальному тлі серед усіх розглянутих методів виділяються методи, які ґрунтуються на аналізі загальної структури геометричного представлення символу. Такі методи складніше в реалізації і, можливо, швидкодія таких методів залежить від глибини і складності аналізу графічного представлення символу. Проте, такі методи мають набагато більш високою точністю розпізнавання, а їх логіка набагато зрозуміліше для людського сприйняття.

Ще однією перевагою алгоритмів, які аналізують структуру геометричного представлення символу, є їх незалежність від того, рукописні це символи або друковані [28]. Робота алгоритму залежить лише від набору шаблонів, з використанням якого буде проводитися порівняльний аналіз поточного зображення.

1.4 Порівняльна характеристика методів розпізнавання символів

Проведемо більш наочне порівняння розглянутих методів розпізнавання символів на зображенні. Для цього сформуємо таблицю з основними характеристиками, перевагами та недоліками розглянутих груп методів, та оцінимо їх перспективність з точки зору подальшої розробки алгоритму розпізнавання, який буде задовольняти нашим вимогам – порівняно невелика

вибірка еталонних зображень, висока точність розпізнавання та порівняно невисока загальна складність алгоритму.

Таблиця 1.4

Порівняння методів розпізнавання символів на зображенні

Група методів	Методи на основі ознакових класифікаторів	Статистичні методи	Методи з використанням структурних складових
Основні принципи	найбільш широке застосування; можна абстрагуватися від деталей процесу порівняння образів; нейронні мережі і машини опорних векторів	Образ аналізується з метою виявлення різних статистичних величин, будуються гістограми, рахуються середні значення і середньоквадратичні відхилення в певних областях. Отримані дані використовуються для виконання класифікації.	На графічному поданні символу виділяються ключові елементи – структурні складові. Вони можуть бути представлені, наприклад, графічними примітивами або послідовністю дій оператора при зображенні символу.
Переваги	Порівняно неважка реалізація; Велика кількість готових бібліотек	Більша наочність для людського сприйняття і можуть бути корисні в процесі налагодження або поліпшення алгоритму.	Висока точність при невеликій кількості еталонних образів; можливість самостійно розробити структурні складові символів
Недоліки	Відсутність можливості надати зручний для сприйняття людиною аргументовану звіт про те, чому образ віднесений саме до того чи іншого класу; сильна залежність точності результату від розміру вибірки	Порівняно низька точність	Трудомісткість розробки

Таблиця 1.4 продовження

Порівняння методів розпізнавання символів на зображенні

Застосовність до нашого завдання	Чи не вирішує проблему величини навчальної вибірки; Не дозволяє в достатній мірі дослідити шляхи оптимізації; Відсутність принципової наукова новизна	Важко досягти необхідної точності на всіх групах символів	Опрацювання методу дозволить вирішити поставлену завдань; Правильно підібрані структурні елементи попередньо обробленого зображення дозволять істотно зменшити ресурсомісткість самого етапу розпізнавання і кількість еталонних образів (аналога навчальної вибірки); запропонований метод дозволяє з високою точністю визначати клас зображення рукописного символу без використання будь-яких ознакових класифікаторів: штучних нейронний мереж, радіально-базисних мереж, що самоорганізуються карт Кохонена, машин опорних векторів і т.д.

Як видно із таблиці, найкраще для нашої мети підходить третя група методів – заснованих на виділенні структурних складових. Для обраної групи методів, як і для інших розглянутих, надзвичайно важливим етапом є попередня обробка зображення. Цей етап більш детально буде розглянуто пізніше.

1.5 Огляд систем оптичного розпізнавання символів

Засновуючись на розглянутих вище методах розпізнавання символів на сьогодні існує три основні типи систем, які реалізують вищезгадані методи [30]. У технічних системах будь-яке рішення при розпізнаванні тексту приймається неоднозначно, а шляхом послідовного висунення і перевірки гіпотез і залучення як знань про сам досліджуваний об'єкт, так і загального контексту. Цілісний опис класу об'єктів сприйняття відповідає двом умовам: по – перше, всі об'єкти даного класу задовольняють цьому опису, а по-друге, жоден об'єкт іншого класу не задовольняють йому. Наприклад, клас зображень букви "А" повинен бути описаний так, щоб будь-яке зображення літери "А" в нього потрапляло, а зображення всіх інших букв – ні. Такий опис має властивість відображуваності,

тобто забезпечує відтворення описуваних об'єктів: еталон букви для системи OCR дозволяє візуально відтворити букву, еталон слова для розпізнавання мови дозволяє вимовити слово, а опис структури пропозиції в синтаксичному аналізаторі дозволяє синтезувати правильна пропозиція. З практичної точки зору відображуваність грає величезну роль, оскільки дозволяє ефективно контролювати якість описів.

Сьогодні відомо три підходи до розпізнавання символів – шаблонний, структурний і просторі ознак. Шаблонний опис простіше в реалізації, однак, на відміну від структурного, він не дозволяє описувати складні об'єкти з великою різноманітністю форм. Саме тому шаблонне опис застосовується для розпізнавання зазвичай лише друкованих символів, в той час як структурний – для рукописних, що мають, природно, набагато більше варіантів накреслення.

1.5.1 Шаблонні системи. Огляд системи Imago OCR

Такі системи перетворюють зображення окремого символу в растрове, порівнюють його з усіма шаблонами, наявними в базі і вибирають шаблон з найменшою кількістю точок, відмінних від вхідного зображення. Шаблонні системи досить стійкі до дефектів зображення і мають високу швидкість обробки вхідних донних, але надійно розпізнають тільки ті шрифти, шаблони яких їм "відомі". І якщо розпізнається шрифт хоч трохи відрізняється від еталонного, шаблонні системи можуть робити помилки навіть при обробці дуже якісних зображень.

Imago OCR. Огляд системи

Imago OCR – це набір інструментів для 2D-розпізнавання хімічної структури [31]. Він містить програму графічного інтерфейсу та утиліту командного рядка, а також задокументований API для розробників. Imago є абсолютно безкоштовним та з відкритим кодом, а також доступним на комерційній основі. Основна частина Imago написана з нуля в сучасній C ++. Він використовує найвідоміші алгоритми оптичного розпізнавання. Це гарантує видатну портативність та продуктивність Imago.

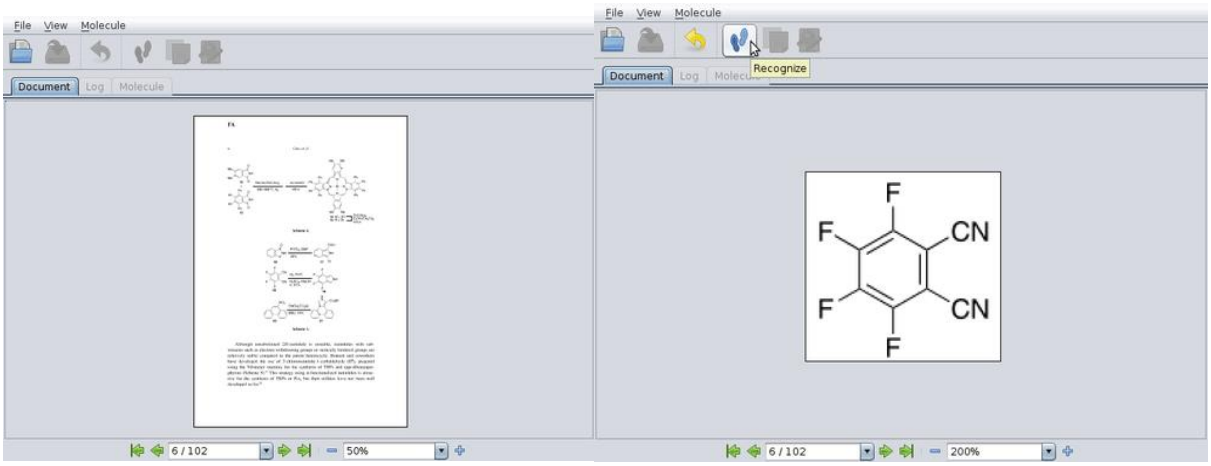


Рисунок 1.12. Користувацький інтерфейс системи Imago OCR

Робота з програмним забезпеченням.

Розширення

Файл imago-gui.jar входить до пакету для кожної архітектури. У ньому немає залежних файлів. Двійкові динамічні бібліотеки витягуються з файлів JAR і завантажуються автоматично.

Інтерфейс

Інтерфейс простий і зрозумілий. Ви можете відкрити зображення (PNG / JPEG / GIF) або документ (PDF / TIFF) і переглянути на вкладці “Документ”.

Ви можете збільшити зображення (або сторінку документа) і вибрати потрібну структуру за допомогою миші. Тоді ви побачите обрану структуру.

Незабаром після натискання кнопки «Розпізнати» ви отримаєте відтворене зображення із розпізнаною молекулою. Ви також можете переглянути журнал програми, щоб дізнатись, що сталося і скільки часу це зайняло. Кнопка «Зберегти молекулу» відкриває діалогове вікно для збереження отриманого молфіла. Кнопка «Відкрити ескіз» відкриває результуючу molfile у зовнішній програмі, яку можна вказати.

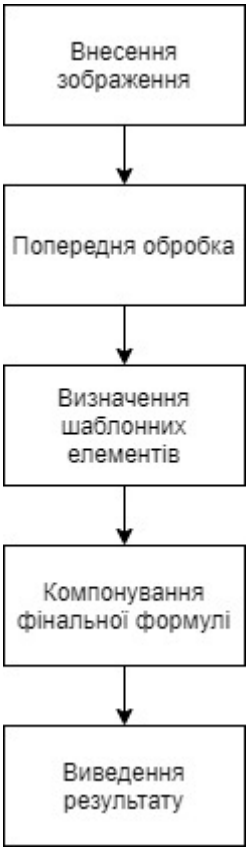


Рисунок 1.13. Загальна структурна схема системи Imago OCR

1.5.2 Системи, засновані на ознакових класифікаторах. Огляд системи Tesseract OCR

У таких системах усереднене зображення кожного символу представляється як об'єкт в n-вимірному просторі ознак. Тут вибирається алфавіт ознак, значення яких обчислюються при розпізнаванні вхідного зображення. Отриманий n-мірний вектор порівнюється з еталонними, і зображення відноситься до найбільш підходящому з них. Ознакові системи не відповідають принципу цілісності. Необхідна, але недостатня умова цілісності опису класу об'єктів (в нашому випадку це клас зображень, що представляють один символ) полягає в тому, що опису повинні задовольняти всі об'єкти даного класу і жоден з об'єктів інших класів. Але оскільки при обчисленні ознак втрачається значна частина інформації, важко гарантувати, що до даного класу вдасться віднести тільки <рідні> об'єкти.

Tesseract OCR. Огляд системи

Tesseract – це система оптичного розпізнавання символів (OCR) [32]. Застосовується для перетворення графічних документів у документи PDF або

Word, які можна редагувати / шукати. Це безкоштовне програмне забезпечення з відкритим кодом, яке запускається через інтерфейс командного рядка (CLI). Tesseract вважається одним з найточніших механізмів OCR з відкритим вихідним кодом, який спонсорує Google з 2006 року. Однак, його можливості можуть бути більш обмеженими, ніж комерційне програмне забезпечення, таке як Adobe Acrobat Pro та ABBYY FineReader. Однак, оскільки це програмне забезпечення з відкритим кодом, кожен, хто має знання програмування, може редагувати код, що стоїть за Tesseract, і допомогти йому дізнатися, що вам потрібно робити. Його можна використовувати на комп'ютерах Mac, Windows та Linux.

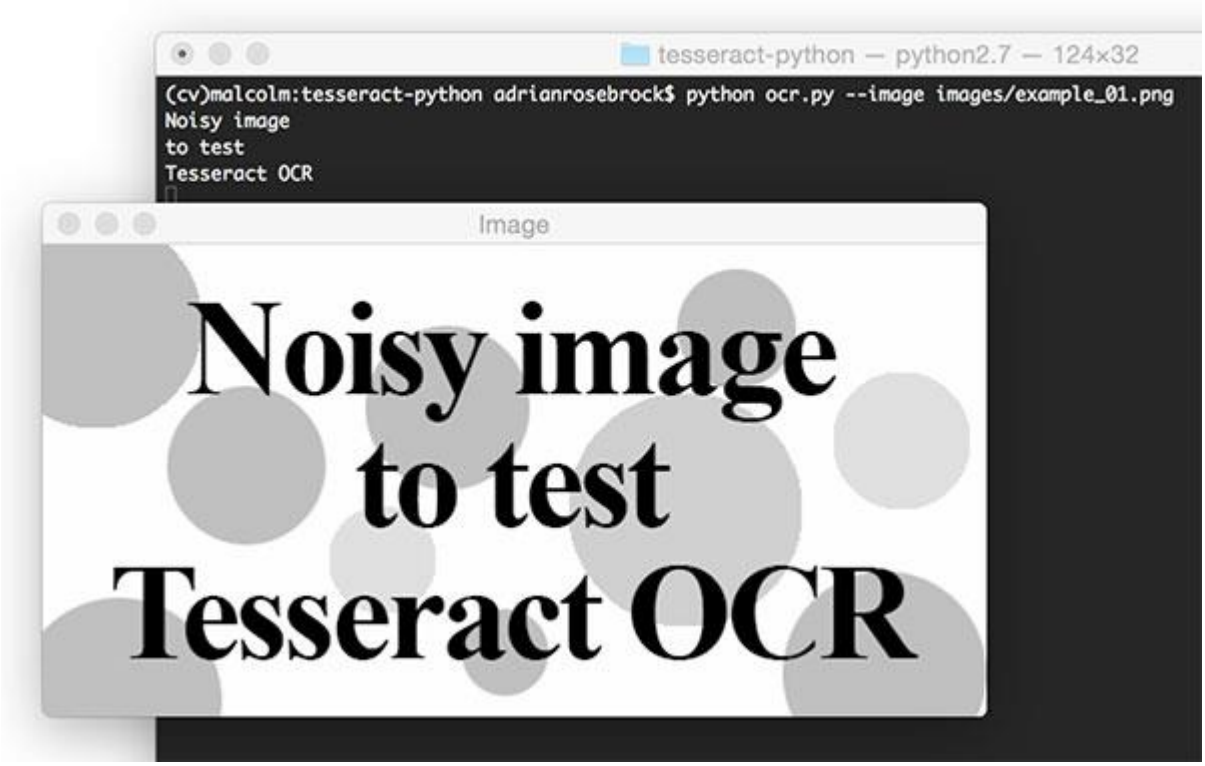


Рисунок 1.14. Приклад роботи системи Tesseract OCR

Робота з програмним забезпеченням.

Користувач вводить у Tesseract заголовок документа, потрібний заголовок та бажаний формат. Tesseract аналізує ці зображення і створює новий документ, який можна шукати у бажаному користувачем форматі. На відміну від іншого програмного забезпечення OCR, ви не можете сканувати щось безпосередньо у Tesseract.

Основні операції OCR в Tesseract:

- Формат зображення (JPG, TIF, PNG тощо) у форматі PDF, Microsoft Word;

- Новий документ відображається в тому ж каталозі, що і початковий документ;
- Система не передбачає власного візуального інтерфейсу, працює через інтерфейс командного рядка.

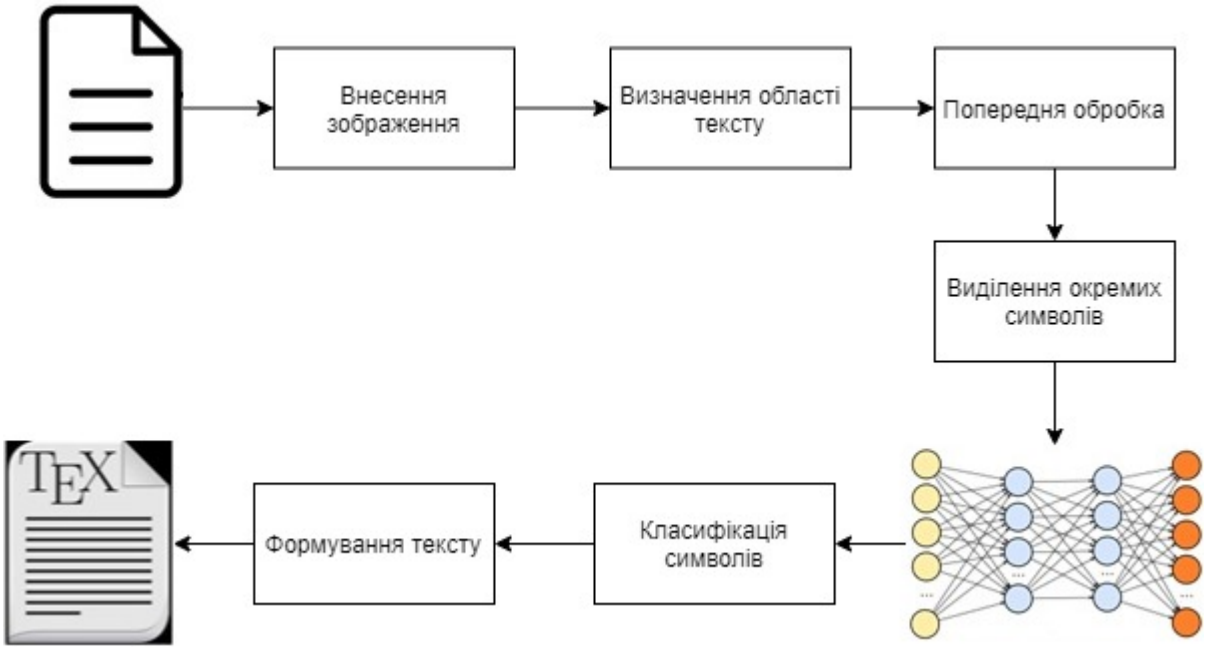


Рисунок 1.15. Загальна структурна схема системи Tesseract OCR

1.5.3 Структурні системи. Огляд системи Abbyy FineReader

У таких системах об'єкт описується як граф, вузлами якого є елементи вхідного об'єкта, а дугами – просторові відносини між ними. Система реалізують подібний підхід, звичайно працюють з векторними зображеннями. Структурними елементами є складові символ лінії. Так, для букви "р" – це вертикальний відрізок і дуга.

До недоліків структурних систем слід віднести їх високу чутливість до дефектів зображення, що порушує складові елементи. Також векторизация може додати додаткові дефекти. Крім того, для цих систем, на відміну від шаблонних і ознакових, до сих пір не створені ефективні автоматизовані процедури навчання. Тому, наприклад, для системи Fine Reader структурні описи довелося створити вручну.

Abbyy FineReader. Опис системи.

ABBYY FineReader – програма для оптичного розпізнавання символів, розроблена російською компанією ABBYY [32].

Програма дозволяє переводити зображення документів (фотографії, результати сканування, PDF-файли) в електронних редактованих форматах. Зокрема, в Microsoft Word, Microsoft Excel, Microsoft Powerpoint, Формат тексту, HTML, PDF / A, PDF, CSV та текстові (звичайний текст) файли. Начину з 11 версій файлів можна зберегти у форматі djvu. Версія 14 підтримує розподіл тексту на 192 мовах і має вбудовану перевірку орфографії для 48 із них. Помимо прочих мов, початкова з версією 10, надається підтримка старого орфографії російського мови, а також з версіями 12 з'явилася словацька підтримка для цієї мови.

Програма доступна для Windows та macOS. Ядро FineReader без графічного інтерфейсу доступний для Linux. У світі більше 20 мільйонів користувачів ABBYY FineReader.

Робота з програмним забезпеченням.

Редактирование тексту в PDF-документах

Ви можете переміщувати текстові блоки та зображення, додавати нові строки або абзаци тексту, редагувати вміст таблиці, змінювати порядок елементів та розмір сторінок прямо в PDF, без додаткової конвертації документів у редактований формат.

Коли ви відкриєте документ, ABBYY FineReader PDF 15 автоматично визначає, доступний пошук за тексту. Якщо немає – програма запускає розпознавання тексту (ABBYY OCR), щоб відкрити доступ до інформації, що містить документи.

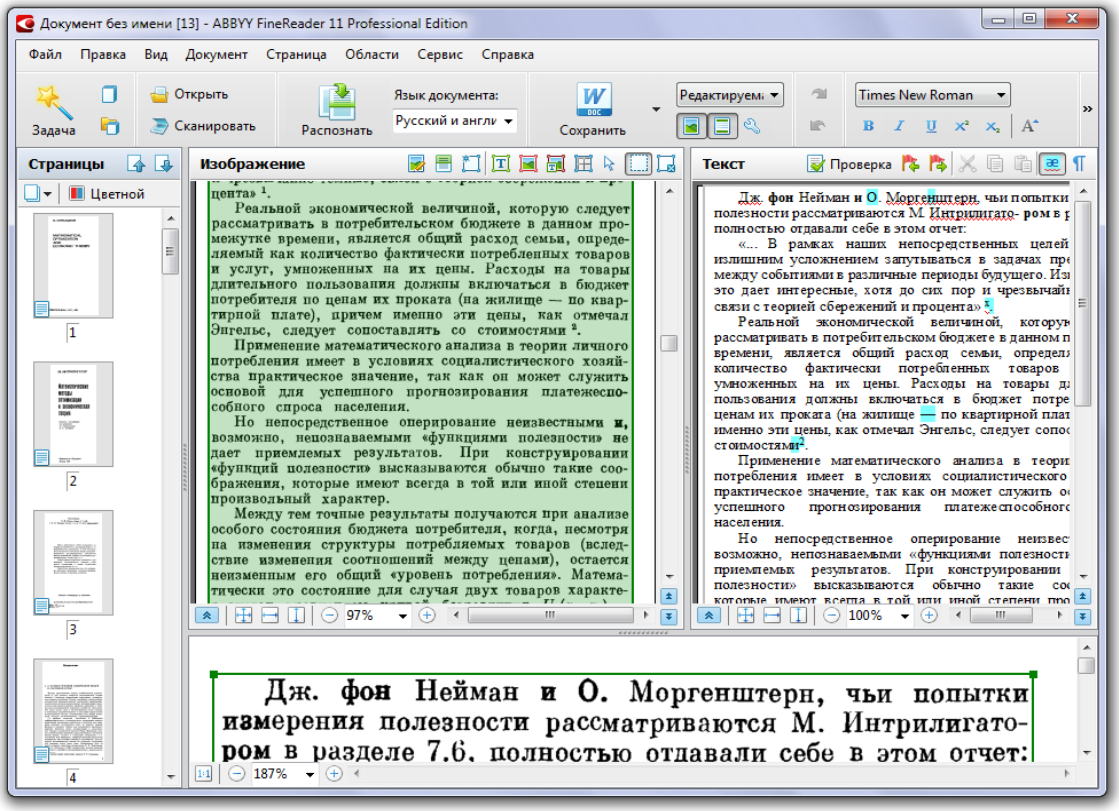


Рисунок 1.16. Приклад роботи системи Abbyy FineReader

Рисунок 1.17. Загальна структурна схема системи ABBYY FineReader

1.6 Порівняльна характеристика систем розпізнавання символів

Розглянемо більш детально переваги та недоліки різних систем оптичного розпізнавання символів на реальних прикладах, та порівняємо їх ефективність.

Таблиця 1.5

Детальне порівняння OCR систем

Назва системи	Imago OCR	FineReader	Tesseract
Тип розпізнавання	Статистичне розпізнавання	Розпізнавання методом виділення структурних складових	Розпізнавання на основі ознакових класифікаторів
Основні відомості	Imago OCR – це набір інструментів для 2D-розпізнавання хімічної структури. Він містить програму графічного інтерфейсу та утиліту командного рядка, а також задокументований API для розробників. Основний компонент Imago написаний з нуля на C ++.	ABBYY FineReader – дозволяє переводити зображення документів (фотографій, сканованих, PDF-файлів) в електронні редаговані формати. Зокрема, в Microsoft Word, Microsoft Excel, Microsoft Powerpoint, Rich Text Format, HTML, PDF / A, searchable PDF, CSV і текстові (plain text) файли. Починаючи з 11 версії файли можна зберігати в форматі djvu.	Tesseract – це механізм розпізнавання тексту з відкритим кодом, доступний за ліцензією Apache 2.0. В основі лежить LSTM нейронна мережа. Tesseract можна використовувати безпосередньо через командний рядок або за допомогою API для вилучення друкованого тексту із зображень. Він підтримує широкий спектр мов. Tesseract не має вбудованого графічного інтерфейсу.
Точність	92%	97%	93%
Об’єм еталонних зображень	середній	середній	великий
Переваги	Висока точність у випадках чітко заданого словника	Висока надійність розпізнавання	Відкритий код, багато прикладів реалізації
Недоліки	Необхідність дотримуватися “шаблонного” шрифту.	Чутливість до якості зображення, потреба в якісній попередній обробці зображення	Великий об’єм навчаючої вибірки, неможливість якісно дослідити фактори впливу на точність розпізнавання

Усі розглянуті системи, незалежно від типу методу, що лежить в їх основі, потребують певної попередньої обробки зображення у випадку його зашумленості або нестандартного кута нахилу. Найвищу точність можна побачити у системі, заснованій на методі виділення структурних складових. Однак для реалізації подібної системи обов'язково потрібно використовувати надійні методи попередньої обробки зображення. Також вирішення потребує задача зменшення вибірки еталонних зображень для настройки системи. Більш детально це буде розглянуто у наступному розділі.

Висновки до першого розділу

У цьому розділі проведено огляд систем, методів і алгоритмів розпізнавання тексту на зображенні. Встановлено, що велика частина розроблених на сьогодні алгоритмів потребує великої кількості еталонних зображень для стадії навчання або пошуку залежностей і закономірностей. У процесі дослідження було прийнято рішення здійснювати розробку системи розпізнавання тексту на зображенні, заснованої на алгоритмі виділення структурних складових.

У більшості із розглянутих систем та алгоритмів розпізнавання символів на зображенні використовується попередня обробка вихідних графічних представлень символів. Наприклад, використовуються вейвлети і курвлети для видалення шумів на зображеннях. Для зображень, які пройшли стадію бінаризації, використовується скелетизація (утоншення) або морфологічні операції для видалення різних дефектів, допущених при друку документа, який був відсканований. Основною перевагою такого підходу є можливість істотно зменшити кількість зображень для навчання класифікатора або настройки алгоритму.

Розглянуті OCR системи не виконують в повній мірі поставлене завдання – розпізнавання рукописного тексту з високою точністю та за умови невеликої вибірки.

Прийнято рішення про необхідність розробки системи розпізнавання тексту на зображенні, заснованої на алгоритмі виділення структурних складових з використанням методу структурних складових.

РОЗДІЛ 2. АНАЛІЗ МЕТОДІВ ПОПЕРЕДНЬОЇ ОБРОБКИ ЗОБРАЖЕННЯ ДЛЯ ПОБУДОВИ СИСТЕМИ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННІ, ЗАСНОВАНОЇ НА СТРУКТУРНИХ МОДЕЛЯХ СИМВОЛІВ

Попередню обробку вихідного зображення можна розділити на три етапи:

1. Підвищення контрастності вихідного зображення.
2. Бінаризація отриманого на попередньому етапі зображення.
3. Скелетизація бінарного зображення.

Після послідовного виконання цих трьох етапів попередньої обробки, отримане скелетизоване бінарне зображення буде вихідним для алгоритму розпізнавання символів.

2.1 Застосування гістограмної еквалізації для збільшення контрастності зображення

При вирішенні задачі розпізнавання символу, як правило, в якості вхідного зображення використовується сканований документ або елемент фотозображення [33, 34]. У таких випадках процес розпізнавання може ускладнюватися різними факторами: неоптимальна настройка пристрою, що виконує зйомку або сканування, поганий рівень освітленості, неоднорідність матеріалу, на якому виконано накреслення символу і т. д. У більшості з таких випадків на оригінальному документі символи можуть погано відрізнятися від фону. Це пояснюється низькою контрастністю зображення.

Для підвищення контрастності зображення досить скористатися загальновідомим методом гістограмної еквалізації, який реалізований в більшості програмних бібліотек комп'ютерного зору. Суть цього методу полягає в тому, щоб інтегральна гістограма зображення була якомога ближче до лінійної функції.

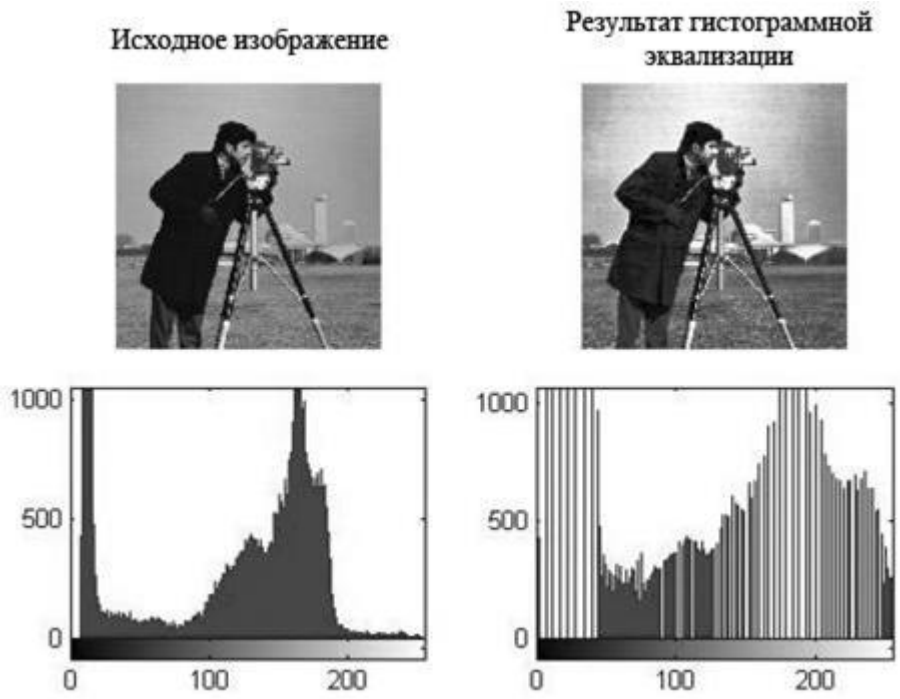


Рисунок 2.1. Візуалізація принципу роботи гістограмної еквалізації

Побудова гістограми зображення виконується з використанням одноразового проходу по всім пікселям зображення і по всіх рівнях яскравості, представленим на ньому [35, 36].

Перетворити гистограмму зображення в його інтегральну гистограмму можна за допомогою одного проходу по всім допустимим значенням яскравості для цього зображення:

$$H_0^{`} = H_0,$$
$$H_i^{`} = H_{i-1}^{`} + H_i (i \geq 1)$$

$H_0^{`}$ – значення гістограми зображення,

$H_i^{`}$ – значення його інтегральної гістограми.

Для збільшення контрастності зображення необхідно повторити лінійний прохід по всім пікселям вихідного зображення, замінивши значення яскравості кожного пікселя.

Як правило, кількість допустимих рівнів яскравості на зображенні набагато менше, ніж кількість пікселів, отже, можна вважати, що алгоритм має обчислювальну складність O(P), де P – кількість пікселів на зображенні.

Алгоритм, що володіє таким високим швидкодією, не зробить істотного впливу на швидкодію процесу розпізнавання в цілому, однак дозволить

уникнути ускладнень в ході розпізнавання через знижену контрастності вихідного зображення.

2.2 Адаптивна бінаризація вихідного зображення

Для виконання розпізнавання символу необхідно попередньо розділити всі пікселі цього зображення на пікселі, які належать графічного представлення символу, і залишилися пікселі.

Можна вважати, що на зображенні в градаціях сірого пікселі, що належать зображенню символу, істотно відрізняються за середнім рівнем яскравості від пікселів фону. Для визначеності будемо вважати, що світліші пікселі належать фону. У такому випадку для поділу пікселів на два описаних класу зазвичай використовується деяке порогове значення яскравості T. Пікселі з рівнем яскравості вище T покладаються білими (належать фону), що залишилися пікселі – чорними (належать графічного представлення символу). Процес виконання такого розбиття називається бінаризація, отримане в ході цього процесу зображення називається бінаризованим, а вибране значення T – порогом бінаризації. Вибір порогового значення яскравості T є нетривіальним завданням, рішення якої суттєво залежить від конкретного класу зображень.

Існує цілий клас методів адаптивної бінаризації, в якій виконується не тільки поділ пікселів вихідного зображення в залежності від порогового значення T, але і виконується автоматичний вибір цього значення [37].

Для визначення оптимального порогу бінаризації T запропоновано величезна кількість різних алгоритмів. Як правило, такі методи спираються на використання локальної або глобальної гістограми зображення. Одним з найбільш широко використовуваних підходів є підхід, запропонований Оцу.

Метод Оцу грунтується на використанні гістограми зображення. Розглянемо, для визначеності, зображення у відтінках сірого. Його гістограма містить 256 стовпців. Якщо розглядати довільну підгістограмму цієї гістограми, то для неї можна обчислити оцінку дисперсії яскравості. Використовуючи оцінку дисперсії, для певного порогу бінаризації T можна визначити критерій розділимості: SP(T):

$$SP(T) = 1 - \frac{D(0,T) + D(T+1,255)}{D(0,255)} \quad (2.1)$$

Критерій завжди приймає значення з відрізка [0, 1], причому, чим більше це значення, тим краще роздільність розподілу яскравості на два класи щодо порога бінаризації T.

Алгоритм Оцу передбачає обчислення критерію для всіх значень T. Оптимальним значенням порога бінаризації T_{opt} вважається значення T, що відповідає максимальному значенню критерію.

Перевагою даного алгоритму є зрозумілий статистичний сенс і низька обчислювальна складність [38].

Після того, як алгоритм Оцу був використаний для бінаризації вихідного зображення, можна вважати, що всі наступні етапи обробки графічного представлення символу працюють з бінарним зображенням, чорний піксель якого належить графічного представлення символу, а білий – фону.

2.3. Алгоритм скелетизації бінарного зображення для розпізнавання символа

Скелетизація – процес, в ході якого виконується максимальне утоньшення всіх ліній на бінарному зображенні, результатом чого є скелет, в якому всі лінії мають товщину в один піксель.

У задачі розпізнавання символів дуже важливо, щоб в процесі скелетизації не відбувалося втрати інформації про ключові елементи графічного зображення вихідного символу. З іншого боку, для методів, що спираються на використання скелетизованого зображення, важливо, щоб жоден з пікселів, який можна видалити, не порушуючи топології графічного представлення, не був пропущений в ході роботи алгоритму.

2.3.1 Алгоритм скелетизації Зонга-Суня

T.Y. Zhang і C.Y. Suen в 1984 році запропонували власний алгоритм скелетизації бінарних растрових зображень [39]. При асимптотично оптимальній реалізації швидкодія цього алгоритму лінійно залежить від кількості пікселів вихідного зображення.

Ідея алгоритму полягає в послідовному виконанні певного набору дій до тих пір, поки хоча б один піксель зображення змінюється після виконання цих дій.

В ході опису алгоритму найбільш зручним буде вважати, що пікселі, що відносяться до графічного поданням символу, є чорними, що залишилися пікселі – білими. Для кожного пікселя розглядаються його вісім сусідніх з ним пікселів бінарного зображення, які в свою чергу, пронумеровані за схемою, показаної на рис. 2.2.

P9	P2	P3
P8	**P1**	P4
P7	P6	P5

Рисунок 2.2. Порядок нумерації сусідніх пікселів у алгоритмі Зонга-Суня

Для пікселів, які знаходяться на границі, всі неіснуючі сусідні пікселі покладаються білими.

Визначимо A (P1), як кількість переходів від білого пікселя до чорного при циклічному обході сусідів в порядку P2, P3, P4, P5, P6, P7, P8, P9, P2. Також визначимо B (P1) рівною кількості чорних сусідів центрального пікселя P1.

На першому кроці алгоритму потрібно додати тег всім пікселі зображення P1, які одночасно задовольняють наступним вимогам:

- Піксель є чорним і має вісім сусідів;
- Виконується нерівність $2 \leq B(P1) \leq 6$;
- Виконується рівність $A(P1) = 1$;
- Як мінімум один з пікселів P2, P4 або P6 є білим;
- Як мінімум один з пікселів P4, P6 або P8 є білим.

В кінці першого кроку потрібно видалити всі пікселі, які були помічені.

На другому кроці алгоритму потрібно додати тег всі пікселі зображення P1, які одночасно задовольняють дещо іншим списку вимог:

- Піксель є чорним і має вісім сусідів;
- Виконується нерівність;
- Виконується рівність;
- Як мінімум один з пікселів P2, P4 або P8 є білим;
- Як мінімум один з пікселів P2, P6 або P8 є білим.

Аналогічно першому кроку, після другого кроку алгоритму потрібно видалити всі помічені під час цього кроку пікселі.

Перший і другий кроки алгоритму потрібно виконувати до тих пір, поки хоча б один піксель позначений в ході цих дій.

Варто відзначити, що при стандартному підході багаторазовий прохід по вихідному зображенню призведе до обчислювальної складності порядку O(P2). Однак, якщо на кожній ітерації алгоритму, крім першої, перевіряти лише пікселі, хоча б один з сусідів яких змінився минулого ітерації, то обчислювальна складність такого алгоритму визначається, як O(P). Реалізувати такий підхід можна з використанням абстрактного типу даних queue (черга).

2.3.2 Алгоритм скелетизації Ву-Цая

У 1992 році Rey-Yao Wu і Wen-Hsiang Tsai запропонували власний метод скелетизації бінарного растрового зображення, що володіє високою швидкодією.

Автори запропонували підхід, при якому всі пікселі зображення потрібно розглянути лише по одному разу. Для виконання скелетизації передбачається використовувати 14 шаблонів, які наведені на рис.2.3.

У наведених шаблонах символи 'c', '0', '1', 'x' позначають, відповідно, розглянутий піксель, білий піксель, чорний піксель і піксель довільного кольору. Символ 'y' має особливу властивість – хоча б один з таких пікселів, при накладенні шаблона, повинен бути білим.

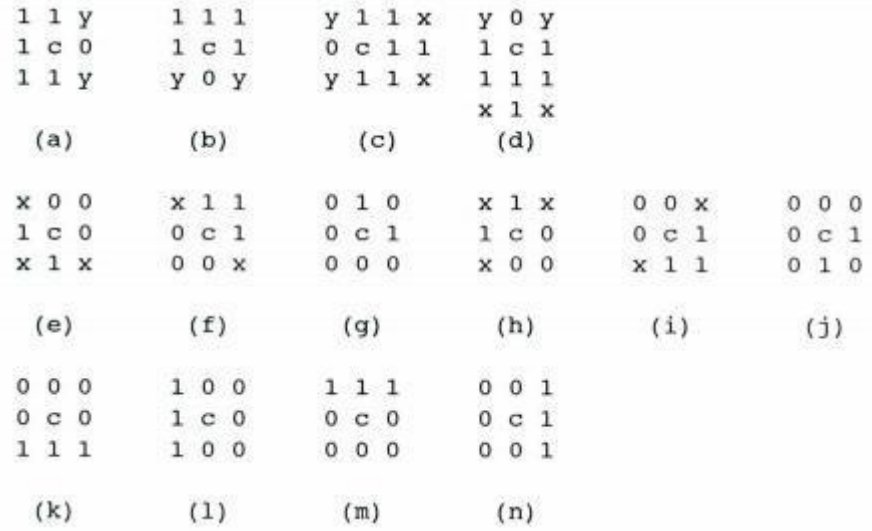


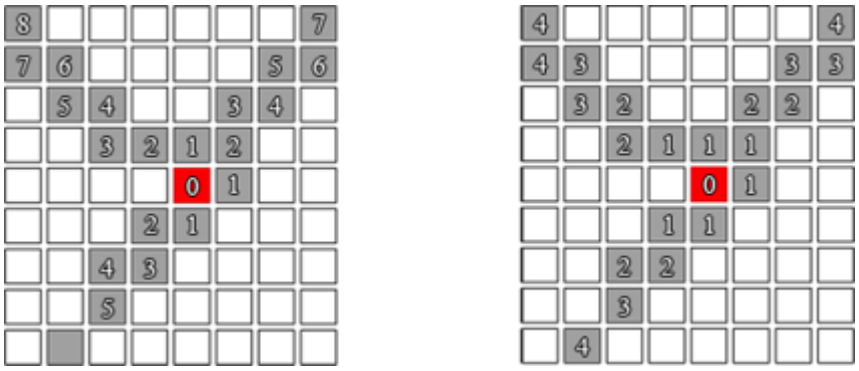
Рисунок 2.3. Шаблони, які використовує алгоритм Ву-Цая

Кожен з наведених шаблонів послідовно застосовується для кожного з пікселів при лінійному проході по всьому пикселам вихідного зображення. Піксель видаляється в разі, якщо піксель, з сусідніми йому пікселями, підходить хоча б під один з описаних шаблонів. Більш того, видалення проводиться до того, як буде розглядатися наступного піксель.

Через те, що всі пікселі зображення розглядаються рівно по одному разу, обчислювальна складність алгоритму становить O(P).

2.3.3 Порівняння та аналіз розглянутих алгоритмів скелетизації

Алгоритм скелетизації Зонга-Суня знаходить широке застосування при вирішенні задачі розпізнавання символів. Основною його перевагою є низька ймовірність видалення важливих елементів графічного зображення в ході процесу скелетизації. У дослідженнях Widiarti A.R. наведено порівняльний аналіз алгоритмів побудови скелетів бінаризованих растрових зображень. У наведеному дослідженні відзначається, що алгоритм Зонга-Суня показав найбільш високу швидкодію і низьку ймовірність видалення важливих елементів графічного зображення в ході процесу скелетизації.



(а) (б)

Рисунок 2.4. Результат обходу алгоритмом Лі одного і того ж бінарного зображення для компонент (а) чотирьох-зв'язності і (б) восьми-зв'язності

Однак, як показали результати дослідження методу Зонга-Суня, одне з його головних достоїнств є і одним з його істотних недоліків. На результуючому зображенні досить часто залишаються пікселі, які можна видалити без порушення топології вихідного графічного зображення. На малюнку 2.5 показаний приклад такого результату застосування алгоритму.

На перший погляд, такого роду недолік не повинен мати істотного впливу на подальшу поведінку алгоритму розпізнавання символів. Однак для більшості методів, що спираються на виділення структурних складових, велика кількість не до кінця скелетизованих ділянок може виявитися критичною. Наприклад, для аналізу структури графічного накреслення символу багато методів використовують обхід пікселів бінарного зображення за допомогою алгоритму Лі. Даний алгоритм є окремим випадком застосування хвильового алгоритму для компонент чотирьох-зв'язності або восьми-зв'язності в залежності від розв'язуваної задачі.

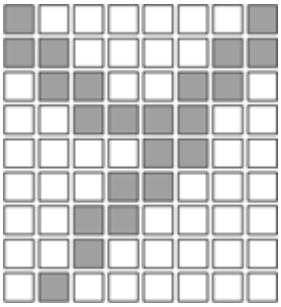


Рисунок 2.5. Приклад не до кінця скелетизованого зображення

На рис. 2.5 представлена візуалізація застосування алгоритму Лі до не до кінця скелетизованого зображення, як для варіанту з компонентами чотирьох-зв'язності, так і для варіанту з компонентами восьми-зв'язності.

Як можна помітити, в обох випадках застосування алгоритму Лі до такого роду зображення приводить до деяких недоліків.

У випадку з компонентами чотирьох-зв'язності два кінцевих пікселя були відвідані з різним номером хвилі, хоча в другому випадку кількість кроків, необхідних для досягнення цих пікселів з початкового пікселя, збігається. Як наслідок, номер хвилі слабо пов'язаний з реальною відстанню від початкової точки і залежить від того, наскільки якісно була виконана скелетизації області окремої ділянки зображення. Але набагато істотніше той факт, що один з кінцевих пікселів і зовсім не відвіданий в ході роботи алгоритму через свою недосяжність при переході лише до сусідніх по стороні пікселям.

У випадку з компонентами восьми-зв'язності на зображенні є сусідні з боці пікселі, що мають однаковий номер хвилі. Це говорить про те, що порядок відвідування цих двох пікселів при обході з довільної вершини може бути

недетермінованим. При оцінці топології об'єкта критичним може бути наявність більш ніж одного простого шляху між двома пікселями, хоча бінарне зображення не містить жодного циклу.

Якщо в першому випадку для усунення обох помилок необхідно одночасно і видаляти певні пікселі, і додавати їх, то в другому досить лише видалення зайвих пікселів. Для усунення областей зображення, де скелетизації була виконана в повному обсязі, в даній роботі вперше запропоновано додатково використовувати алгоритм скелетизації Ву-Цая. Після застосування такого підходу небажані ділянки на результуючому зображенні усуваються. Результат обходу алгоритмом Лі після застосування алгоритму Ву-Цая показаний на рис. 2.6.

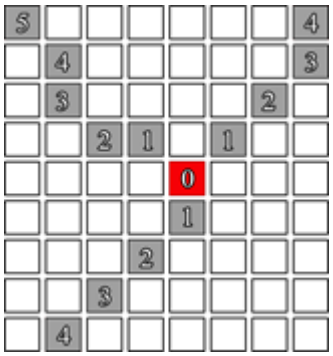


Рисунок 2.6. результат виконання алгоритма Лі після обробки двома алгоритмами скелетизації

Варто зазначити, що самостійне застосування алгоритму Ву-Цая без попередньої обробки алгоритмом Зонга-Суня виявляється недоцільним. Алгоритм Ву-Цая досить часто порушує вихідну топологію об'єкта, видаляючи невеликі елементи графічного представлення символу, наприклад, важливі з точки зору топології виступи на графічному поданні. Можна зробити висновок, що використання лише одного з двох розглянутих алгоритмів скелетизації призведе до втрати інформації про топології символу або до неоднозначності при спробі вилучення структурних складових. У свою чергу, використання спочатку алгоритму Зонга-Суня, а потім алгоритму Ву-Цая призводить до усунення обох типів недоліків.

Однак існують зображення, при обробці яких навіть при такому підході видаляються важливі елементи графічного представлення символу. Приклад такого випадку зображений на рис. 2.7.

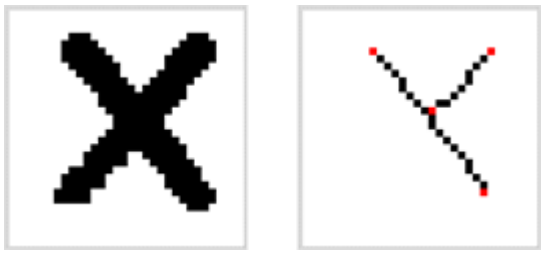


Рисунок 2.7. Приклад некоректної скелетизації послідовним застосуванням алгоритмів Зонга-Суня і Ву-Цая

При детальному аналізі такого роду випадків з'ясувалося, що алгоритм Зонга-Суня іноді теж допускає видалення важливих з точки зору топології символу елементів. Отже, помилка допускається вже першим з двох послідовно застосовуваних алгоритмів.

Як можна помітити, наведене графічне зображення символу «X» містить плоске завершення нижнього лівого елемента. Алгоритмом воно сприймається, як локальне потовщення і поступово віддаляється. Такого роду плоских завершень слід уникати. Видаляючи такі елементи, можна порушити вихідну топологію об'єкта. Отже, необхідно додавати деякі пікселі так, щоб уникати подібних плоских завершень.

Одним із способів рівномірного потовщення бінарного зображення є дилатація – одна з операцій морфологічної обробки зображень. Однак застосування такої операції може привести до з'єднання двох близько розташованих один до одного елементів, тим самим порушити вихідну топологію об'єкта.

Отже, виникає потреба у розробці власного методу скелетизації, на основі оглянутих.

2.4 Алгоритм виділення структурних складових на скелетизованому бінарному зображенні

Виконати виділення структурних складових на скелетизованому бінарному зображенні істотно простіше, ніж на довільному бінарному зображенні [40]. Для такого роду зображень набагато простіше сформулювати правила, за якими визначатимуться найбільш важливі елементи графічного представлення символу і, як наслідок, алгоритми, які дозволять отримати ці елементи.

Людський мозок здатний швидко класифікувати зображений символ, не розглядаючи протягом тривалого часу кожен піксель зображення. Досить швидко виконати класифікацію дозволяє виявлення найбільш важливих точок на графічному зображенні символу. На рис. 2.8 наведені приклади символів, з зазначеними на них найбільш важливими для розпізнавання елементами.

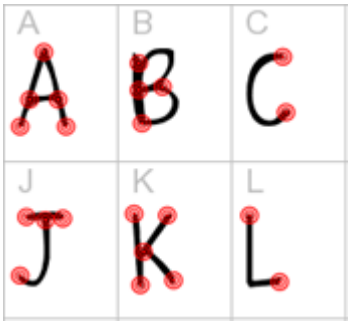


Рисунок 2.8. Приклади букв латинського алфавіту з відміченими ключовими елементами

На наведеному зображенні показані символи латинського алфавіту, на яких відзначені місця закінчення або стику найпростіших графічних примітивів: відрізків і дуг. Якщо несуттєво змінювати з'єднують їх лінії, то ми все одно зможемо дізнатися ці символи.

Виділення таких важливих точок навіть для скелетизованих бінарних зображень залишається нетривіальним завданням [41, 42]. Ще більш нетривіальним завданням є характеристика з'єднують їх елементів. Якщо навчитися отримувати з бінарного зображення інформацію про розташування ключових точок, а також про характеристики з'єднують їх ліній, то на основі цих даних можна побудувати повноцінну структурну модель символу, досить інформативну для його класифікації.

Позначимо терміни, які будуть використовуватися для опису видобутих структурних складових представлення символу, а також для опису алгоритму їх отримання.

Ключові пікселі – чорні пікселі скелетизованого бінарного зображення символу, що відповідають ключовим точкам або вигинів його структурної моделі.

З'єднуюче ребро – структурна складова представлення символу, що описує послідовність чорних пікселів на скелетизованому бінарному зображенні символу, що сполучає два ключових пікселя.

Ключова точка – структурна складова представлення символу, яка є місцем стику одного, трьох або більше з'єднуючих ребер. Також ключовою точкою можна назвати місце стику двох з'єднуючих ребер, направляючі вектори яких, при виборі в якості початкового положення даної ключової точки, розташовані під кутом менше 120 градусів.

Напрямний вектор деякого з'єднуючого ребра може бути обчислений з використанням наступного принципу. Розглянемо з'єднуюче ребро між двома ключовими пікселями, як впорядковану в порядку віддалення від початкового положення послідовність пікселів. В якості початкового положення слід вибрати один з інцидентних ключових пікселів. По черзі проведемо вектори з початкового ключового пікселя в кожен з пікселів шляху. Послідовно підсумуємо в порядку віддалення від початкового ключового пікселя всі вектори зі згасаючими коефіцієнтами: наприклад, перший з множником 1; другий – з множником 0,5; третій – з множником 0,25; і так далі.

Вигин (точка вигину) – структурна складова представлення символу, яка не може бути віднесена до ключових точок, але лінія, що з'єднує два ключових пікселя і проходить через цей піксель, істотно змінює напрямок на ділянці графічного представлення символу, відповідному цьому пікселу. Фактично, вигин є стиком двох з'єднуючих ребер, направляючі вектора яких, при виборі в якості початкового положення даного вигину, розташовані під кутом не менше 120 градусів.

Композитне ребро – послідовність з'єднуючих ребер, що починається і закінчується в ключових точках, яка не містить жодної ключової точки, крім початкової і кінцевої.

Таким чином, чорні пікселі, які не є ключовими пікселями або вигинами, об'єднуються в з'єднуючі ребра, які, в свою чергу, об'єднуються в композитні ребра.

2.5. Алгоритм сегментації рукописного тексту на основі побудови структурних моделей

Вирішувати завдання розпізнавання окремих рукописних символів істотно простіше, ніж вирішувати аналогічне завдання розпізнавання для символів, які є частиною рукописного тексту. В такому випадку необхідно не тільки вирішувати задачу розпізнавання рукописних символів, але і завдання сегментації рукописного тексту – виділення з нього окремих символів і сполучних елементів зображення.

2.5.1. Аналіз вимог до алгоритму сегментації на основі побудови структурних моделей

При виконанні накреслення рукописного тексту почерки розрізняються не тільки способом накреслення окремих символів, а й способом виконання окремих елементів, що з'єднують послідовні літери тексту. Процес сегментації ускладнюється характерними для багатьох почерків спотвореннями накреслень символів при з'єднанні їх графічного представлення з подальшим і попереднім символами.

Для виконання сегментації рукописного тексту необхідно проаналізувати вихідне зображення символу, визначивши приналежність кожної з областей графічного представлення ділянки рукописного тексту до певного символу. Розглянути всі можливі розподілу областей графічного представлення ділянки рукописного символу не представляється можливим зважаючи на величезну кількість різних розподілів навіть для растрового представлення зображення.

Розглянутий раніше алгоритм побудови структурної моделі накреслення окремого символу можна застосувати для побудови структурної моделі накреслення цілого слова. Як і у випадку з окремими символами, в даному випадку необхідно попередньо виконати скелетизації вихідного зображення символу.

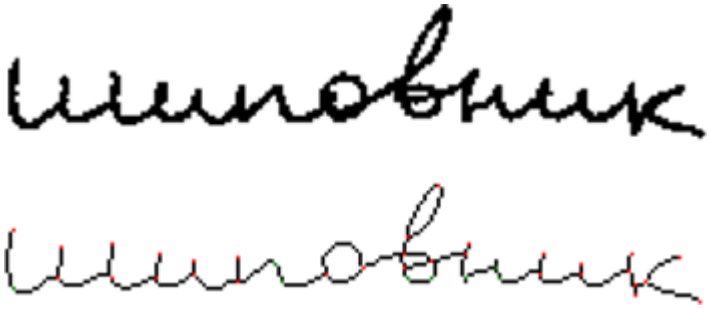


Рисунок 2.9. Приклад зображення окремого слова рукописного тексту і побудованого для нього скелета

В ході роботи алгоритму буде виконано виділення структурних складових, в тому числі ключових точок і вигинів, від місця розташування яких можна відштовхуватися при виборі кордонів, що розділяють області накреслення послідовних символів тексту, що сегментується.

Якщо розглянути приклади структурних моделей окремих слів рукописного тексту, то можна виділити кілька візуальних особливостей цих моделей.

Рисунок 2.10. Приклад структурної моделі окремого слова рукописного тексту

В першу чергу варто відзначити, що більшість розділових ліній між сусідніми буквами слова можна провести через середину з'єднує ребра. Причому будь-яким з кінців такого ребра може бути як ключова точка, так і вигин. При такому способі вибору розділових ліній, частина з'єднувального елемента іноді слід вважати частиною деякого символу, а іноді – слід вважати частиною зображення, яка не відноситься до жодного його символу.

Існують і зображення, для яких можливий варіант сегментації, при якому розділова лінія може бути проведена через ключові точки або вигини. Таке

можливо у випадках, коли при зображенні відсутня необхідність у використанні з'єднувальних елементів. Як можна помітити, не завжди можливо розділити сусідні символи тільки вертикальними лініями. Іноді графічне представлення одного символу може частково знаходитися над або під графічним представленням сусіднього йому символу. Такі випадки можливі не тільки через особливості графічного представлення окремих символів, а й через особливості почерку, пов'язаних з більшим ступенем нахилу букв.

За результатами перерахованих спостережень можна сформулювати список вимог до алгоритму сегментації рукописного тексту:

- можливість враховувати присутність на зображенні сполучних елементів між сусідніми символами;
- можливість віднесення окремих ділянок з'єднувальних елементів до зображенню з'єднуються символів;
- використання невертикальних прямих як розділяють сусідні символи ліній;
- врахування можливого відмінності нахилу накреслень різних символів в одному слові.

Крім того, список вимог може бути доповнений типовими вимогами до алгоритмів обробки зображень:

- поліноміальна залежність часу роботи від вхідних даних;
- поліноміальна залежність обсягу споживаної оперативної пам'яті від вхідних даних.

2.6 Аналіз використання розглянутих алгоритмів попередньої обробки у системах розпізнавання зображення.

Для більшості систем розпізнавання зображення абсолютно необхідним є етап попередньої обробки, який дозволяє привести вхідне зображення до придатного для подальшого розпізнавання стану. Розглянемо більш детально, які саме методи попередньої обробки мають розповсюджене використання на прикладі вже оглянутих у попередньому розділі систем розпізнавання зображення.

Таблиця 2.1

Методи попередньої обробки в системах розпізнавання зображень

Назва системи	Методи попередньої обробки зображення
Imago OCR	Програма розрахована на відносно обмежену кількість можливих символів для розпізнавання (хімічні формули), а отже не потребує надто великої складності попередньої обробки, компенсуючи це попередньо завантаженим “словником” можливих комбінацій символів, що дозволяє прогнозувати наявність елементів, якісно розпізнати які не вдається. Цей підхід, однак, не прийнятний для задачі загального розпізнавання рукописного тексту, адже створення всеосяжного словнику можливих комбінацій елементів (символів) в такому випадку стає занадто затратним, а при неминучій появі непередбаченої комбінації точність розпізнавання відразу суттєво зменшиться.
FineReader	Програма виконує розпізнавання друкованого тексту на 192 мовах. Однак для роботи не з якісними PDF-документами потребує приведення відсканованого зображення до якомога більш чіткого вигляду. Враховуючи значно меншу кількість варіантів написання друкованих символів (у порівнянні з рукописними) у використанні алгоритму скелетизації немає необхідності. Однак без послідовного виконання покращення чіткості (контрасності) та піксельної бінаризації точність розпізнавання може значно впасти, аж до геть неприйнятних результатів при погано відсканованому зображенні, повернутому під певним кутом.
Tesseract	Серед розглянутих систем лише дана система надає можливість якісного розпізнавання саме рукописного тексту, без прив’язки до певного контексту. Не існує чітко задокументованих рекомендацій із оптимальної обробки, однак численними спробами було встановлено, що лише скориставшись методами гістограмної еквалізації, піксельної бінаризації та скелетизації зображення можна досягти точності розпізнавання вищої за 95%.

Жодна із розглянутих систем не відповідає в повній мірі нашій задачі – розпізнаванню рукописних символів за умови порівняно невеликої вибірки еталонних зображень. Однак розглянуті системи чітко підтверджують необхідність застосування алгоритмів попередньої обробки не залежно від використаного в подальшому методу розпізнавання зображення. Оскільки рукописні символи мають значно більше варіантів написання, ніж друковані, то завдання попередньої обробки стає ще важливішим, що можна побачити на прикладі системи Tesseract. В подальшому буде запропоновано оригінальний підхід до покращення алгоритму попередньої обробки зображення.

Висновки до другого розділу

Детально проаналізовано алгоритм попередньої обробки вхідного зображення рукописного символу, що включає стадії гістограмної еквалізації, адаптивної бінаризації, скелетизації та сегментації.

Зроблено порівняння методів скелетизації Зонга-Суня та Ву-Цая. Запропоновано розробити більш оптимальну версію алгоритму скелетизації, засновуючись на послідовному виконанню алгоритмів Зонга-Суня та Ву-Цая

Аналітично розглянуто алгоритм побудови структурної моделі рукописного символу на основі виділених структурних складових: ключових точок, вигинів, з'єднуючих і композитних ребер.

Сформовано вимоги до алгоритму, попередньої обробки зображення та розпізнавання символів на зображенні.

Розглянуто представлені раніше OCR системи з точки зору використання у них алгоритмів попередньої обробки зображень. Зроблено висновок про збільшення важливості етапу попередньої обробки для задачі розпізнавання рукописних символів у порівнянні із задачею розпізнавання друкованих символів.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННІ

У попередньому розділі було досліджено послідовні етапи обробки зображення для подальшої класифікації. Було виявлено певні недоліки, які можуть суттєво вплинути на якість кінцевого розпізнавання. Зокрема, оптимізації вимагає алгоритм скелетизації бінаризованого зображення. Окремих зусиль вимагає розробка власного методу виділення структурних складових на скелетизованому зображенні. Цей розділ присвячено розробці власної версії алгоритму розпізнавання символів на основі виділення структурних складових.

3.1 Розробка і опис структурної схеми системи розпізнавання зображення на основі побудови структурної моделі

Розглянемо основні елементи, з яких повинна складатися розроблювана система. На основі алгоритмів, розглянутих у попередньому розділі, потрібно передбачити сховище для збереження сформованих структурних моделей символів. В подальшому, результат сканування рукописного тексту буде порівнюватися із заранє збереженими структурними моделями зображення. Отже, робота системи має бути поділена на два етапи – внесення (формування) еталонних форм, та розпізнавання тексту шляхом порівняння із раніше введеними формами. Варто також зазначити, що і для етапу формування моделей, і для етапу розпізнавання можна використовувати один і той самий алгоритм попередньої обробки зображення. На рисунку 3.1 зображено загальну структурну схему розроблюваної системи.

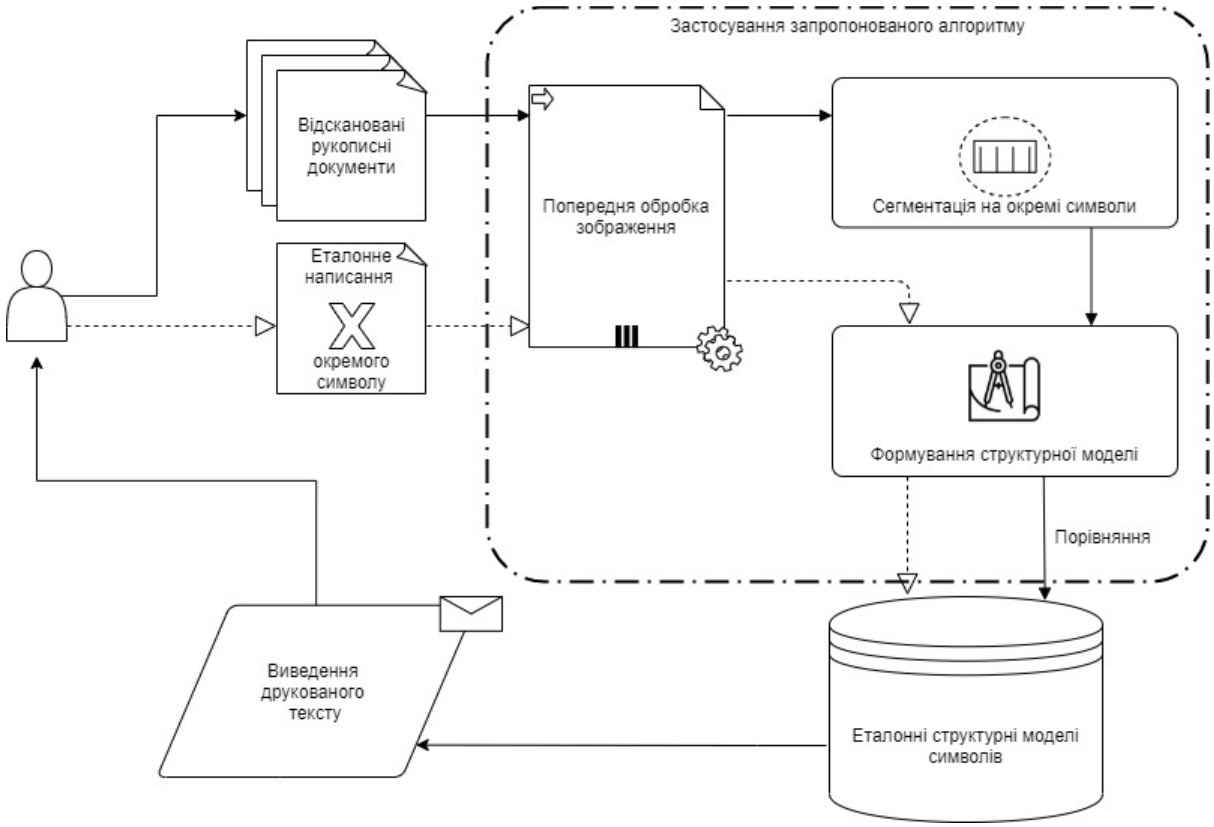


Рисунок 3.1. Загальна структурна схема розроблюваної системи розпізнавання тексту на зображенні

Як видно із структурної схеми, користувачу має бути доступно два шляхи: внесення еталонних зображень окремих символів для формування еталонних

моделей, та внесення відсканованих зображень рукописного тексту для його розпізнавання. У другому випадку до загального алгоритму попередньої обробки в кінці додається етап сегментації слів на зображенні на окремі символи. В результаті формується структурна модель символа, яка в одному випадку вноситься в сховище як еталонна, а в другому – порівнюється з уже внесеними.

3.2 Покращений алгоритм скелетизації.

Існують зображення, при обробці яких при послідовному використанні алгоритмів скелетизації Зонга-Суня та Ву-Цая видаляються важливі елементи графічного представлення символу. Приклад такого випадку зображений на рис. 3.1

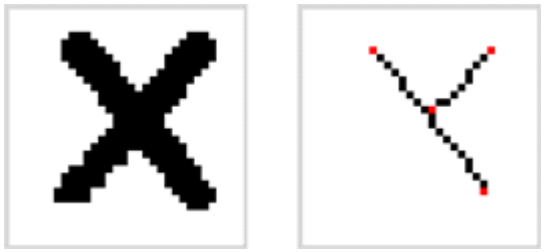


Рисунок 3.1. Приклад некоректної скелетизації послідовним застосуванням алгоритмів Зонга-Суня і Ву-Цая

При детальному аналізі такого роду випадків з'ясувалося, що алгоритм Зонга-Суня іноді теж допускає видалення важливих з точки зору топології символу елементів. Отже, помилка допускається вже першим з двох послідовно застосовуваних алгоритмів.

Як можна помітити, наведене графічне зображення символу «X» містить плоске завершення нижнього лівого елемента. Алгоритмом воно сприймається, як локальне потовщення і поступово віддаляється. Такого роду плоских завершень слід уникати. Видаляючи такі елементи, можна порушити вихідну топологію об'єкта. Отже, необхідно додавати деякі пікселі так, щоб уникати подібних плоских завершень.

Одним із способів рівномірного потовщення бінарного зображення є дилатація – одна з операцій морфологічної обробки зображень. Однак застосування такої операції може привести до з'єднання двох близько

розташованих один до одного елементів, тим самим порушити вихідну топологію об'єкта.

У даній роботі запропонована власна операція усунення плоских закінчень, що має схожість з морфологічною операцією дилатації. Для кожного білого пікселя розглянемо пікселі, які межують з ним по стороні. Якщо кольори чотирьох сусідніх пікселів не утворюють одну з чотирьох комбінацій наведених на рис. 3.2, то піксель стає чорним.

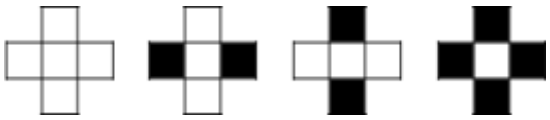


Рисунок 3.2. Комбінації сусідніх пікселів, які не потребують перефарбовування центрального пікселя в чорний колір

Передобробка з використанням такої операції дозволяє уникнути плоских закінчень елементів бінарних зображень, які не потребують вилучень. Для раніше наведеного прикладу символу застосування такої операції призводить до істотного поліпшення результату.

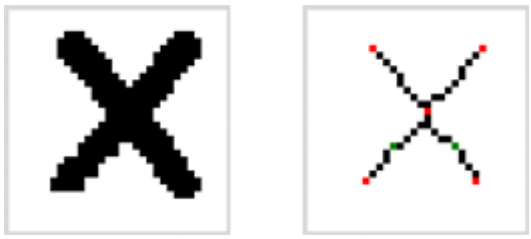


Рисунок 3.3. Результат виконання запропонованого алгоритму скелетизації

Як можна помітити, застосування запропонованої операції до бінаризованого зображення дозволяє уникнути описаної проблеми. Подальше дослідження показало, що для всіх розглянутих в ході експериментів випадків запропонований підхід до скелетизації не порушує вихідної топології символу.

Таким чином, запропонований підхід до виконання скелетизації складається з трьох послідовних етапів:

1. Попередня обробка бінарного зображення з метою усунення плоских закінчень елементів графічного представлення.

2. Застосування алгоритму скелетизації Зонга-Суня.

3. Застосування алгоритму Ву-Цая для усунення ділянок з незакінченою скелетизації на отриманому зображенні.

3.3 Вдосконалення алгоритму виділення ключових пікселів та вигинів на скелетизованому та бінаризованому зображенні

За допомогою аналізу скелетизованих зображень набору рукописних цифр MNIST був визначений набір шаблонів, за якими можна визначити пікселі, які є ключовими або вигинами.

В першу чергу до таких пікселям можна віднести всі пікселі, кількість сусідів у яких не дорівнює двом. Якщо у пікселя немає сусідів, то він є ізольованим. Якщо у пікселя один сусід, то такий піксель є точкою закінчення будь-якого елементу графічного накреслення символу. Якщо ж кількість сусідів у пікселя більше двох, то такий піксель, швидше за все, є точкою з'єднання декількох елементів графічного зображення.

Складніше аналізувати піксель, який має два сусіда. В такому випадку піксель може бути частиною з'єднує ребра або точкою зчленування двох елементів графічного зображення. Експериментально було визначено набір шаблонів для визначення пікселів, які є ключовими або вигинами.

Можна виділити першу групу шаблонів сусідніх пікселів, яка завжди відповідає ключовому пікселю або вигину (див. рис. 3.4), назвемо цю групу шаблонів α-група.

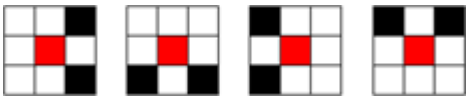


Рисунок 3.4. α-група шаблонів сусідніх пікселів

Для деяких шаблонів сусідніх пікселів недостатньо інформації лише про їх взаємне розташування. Необхідно також врахувати, як розташовані сусіди безпосередніх сусідів центрального пікселя. Виділимо деяку групу таких шаблонів сусідніх пікселів, позначивши її як β-групу:

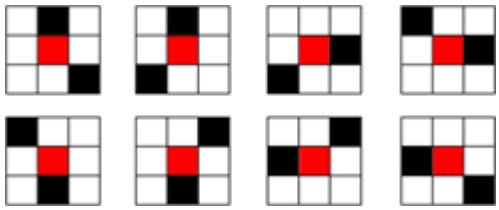


Рисунок 3.5. β-група шаблонів сусідніх пікселів

Як можна помітити, такі ділянки бінарного зображення можуть відповідати, як ділянці з'єднує ребра, так і точці стику декількох сполучних ребер. Експериментально було виявлено закономірність, згідно з якою тип центрального пікселя може бути визначений, виходячи з положення пікселів, які є сусідніми для сусідів даного пікселя. Приклади випадків, коли шаблон β-групи відповідає ключовій точці або точці вигину показані на рис. 3.6:

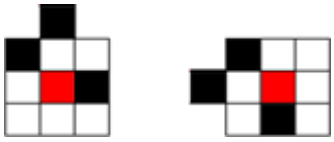


Рисунок 3.6. Приклади випадків, коли шаблон β-групи не є частиною з’єднуючого ребра

Як можна помітити, один сусідній піксель одного з сусідів центрального пікселя є ключовим для визначення типу ділянки бінарного зображення. Однак можна виділити ще одну групу шаблонів сусідніх пікселів, яка, як і β-група, характеризується залежністю не тільки від безпосередніх сусідів центрального пікселя. Назвемо таку групу шаблонів γ-групою. Формально, така група складається лише з двох досить тривіальних шаблонів (див. Рисунок 3.7), які, на перший погляд, не можуть відповідати нічому, крім ділянки з'єднує ребра.

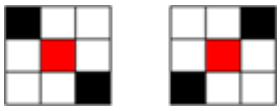


Рисунок 3.7. γ-група шаблонів сусідніх пікселів

Таким чином, були виділені три групи шаблонів розташування сусідніх пікселів: α-група, β-група і γ-група. Пікселі, які їм належать можуть відповідати, як ключовим пікселям, так і вигинам.

3.4 Алгоритм поділу ключових пікселів і вигинів

Фактично, скелетизоване бінарне зображення можна представити у вигляді графа. Чорні пікселі цього зображення в такому поданні є вершинами графа, а ребрами вони з'єднані з усіма чорними пікселями, що межують з ними по стороні або кутку. Таким чином, кожна з вершин такого графа може мати не більше восьми інцидентних ребер.

Запуск обходу такого графа в ширину одночасно з усіх вершин, відповідних пикселям α-, β- і γ-груп можна вважати окремим випадком використання алгоритму Лі для бінарного зображення. В ході роботи алгоритму від кожної з початкових вершин буде розходитися, так звана, хвиля. Дві зустрічні хвилі можуть зустрітися на середині з'єднує ребра, в такому випадку, кожна з хвиль може носити інформацію про вершині, з якої вона розпочата, а також про кількість пікселів на шляху від цієї вершини до місця зустрічі цих хвиль.

Можна заздалегідь визначити два правила, за якими однозначно визначаються пікселі, які є ключовими, а не вигинами. Вершина відповідає ключовому пікселю, якщо:

- має інцидентну йому петлю;
- має кількість вихідних з'єднуючих ребер відмінну від двох.

У решті випадків класифікація вершин вимагає більш ретельного аналізу. До решти випадків можна віднести лише вершини, що мають рівно два вихідних з'єднувальних ребра. Так як вихідних ребер всього два, кут між напрямками двох цих ребер відіграє визначальну роль при класифікації типу пікселя. Якщо кут між двома напрямними векторами вихідних з деякою вершини з'єднуючих ребер не менше 120 градусів, то можна вважати, що ця вершина відповідає вигину.

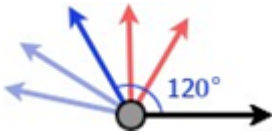


Рисунок 3.8. Кут між напрямними векторами двох вихідних з'єднуючих ребер

На рисунку 3.8 чорним кольором показаний один з направляючих векторів деякої вершини, синім – напрямки другого вектора, при яких дана вершина відповідає вигину, червоним – напрямки другого вектора, при яких дана вершина відповідає ключовому пікселю.

Для того щоб визначити спрямовуючий вектор деякого з'єднуючого ребра можна скористатися наступним принципом. Розглянемо з'єднуюче ребро між двома ключовими пікселями як впорядковану в порядку віддалення від початкової точки послідовність пікселів. По черзі проведемо вектори з початкового ключового пікселя в кожен з пікселів шляху. Послідовно

підсумуємо в порядку віддалення від початкового ключового пікселя всі вектора: перший з множником рівним одиниці; другий – з множником 0,5; третій – з множником 0,25; і так далі. Іншими словами, кожен наступний вектор входить в результуючу суму з множником, вдвічі менше, ніж попередній.

Таким чином, для кожної пари, що складається з вершини і інцидентності їй з'єднує ребра, з використанням описаного методу зваженого підсумовування визначається спрямовує вектор. Так як множник зменшується в геометричній прогресії, пікселі, досить віддалені від початкового, не роблять майже ніякого впливу на напрямок отриманого вектора.

Спираючись на розглянуті твердження, можна сформулювати алгоритм поділу ключових точок і вигинів.

Для пошуку множини ключових пікселів і множини пікселів-вигинів необхідно виконати наступні кроки:

1. Виділимо на бінарному скелетизованому зображенні все пікселі відповідні α-, β- і γ-групам, позначимо всі ці пікселі, як ключові.
2. Запустимо алгоритм Лі одночасно з усіх ключових пікселів.
3. Якщо в результаті роботи алгоритму Лі виявлено, що ті з пікселів, які на цей момент вважалися ключовими, насправді є вигинами, позначимо їх як вигини і повернемося до кроку 2.
4. Якщо в результаті роботи алгоритму Лі не виявлено нових вигинів, алгоритм завершено.

Важливо зауважити, що для кожного наступного запуску алгоритму Лі множина ключових пікселів, з яких будуть одночасно запускатися хвилі, буде зменшуватися. Це обумовлено тим, що за результатами попереднього запуску алгоритму, деякі ключові пікселі можуть стати вигинами, однак вигини вже не можуть знову стати ключовими пікселями. Отже алгоритм має обмежену кількість ітерацій.

Оцінити обчислювальну складність такого алгоритму можна як $O(P \cdot K)$, де K – кількість пікселів бінарного скелетизованого зображення, відповідних α-, β- і γ-групам, а P – кількість чорних пікселів бінарного скелетизованого зображення. Для реальних скелетизованих бінарних зображень величина P

набагато менше, ніж H · W, де W і H – ширина і висота зображення відповідно. Величина K для реальних зображень набагато менше P і рідко перевищує 15.

3.5 Алгоритм виділення композитних ребер

Раніше були розроблені алгоритми для виділення ключових точок і вигинів на скелетизованому бінарному зображенні. Для побудови повноцінної структурної моделі також необхідно виділити з'єднують ребра і об'єднати їх в композитні, які є послідовністю з'єднують ребер від одного ключового пікселя до іншого.

Для того щоб сформулювати алгоритм виділення композитних ребер на оригінальному документі, необхідно попередньо зауважити, що ніякої вигин не належить більш ніж одному композитного ребру або петлі. Це твердження явно випливає з того факту, що на шляху між двома вершинами, відповідними сусіднім ключовим пикселям, все вершини відповідають вигинів і можуть мати лише два інцидентних з'єднують ребра.

Так як будь-який піксель, відповідний вигину, належить тільки одному композитного ребру, відвідування в порядку обходу алгоритмом Лі цього пікселя не викликає неоднозначності при виборі подальшого напрямку обходу. Таким чином, при запуску алгоритму Лі одночасно з усіх ключових пікселів, пікселі, які є вигинами, не можуть вплинути на детермінованість обходу. В такому випадку дві хвилі, запущені з двох ключових пікселів, пов'язаних композитним ребром, обов'язково зустрінуться десь між двома цими пікселями. Якщо використовувати раніше розглянутий спосіб визначення направляючого вектора хвилі і підрахунку кількості відвіданих цією хвилею пікселів, можна сформулювати алгоритм виділення композитних ребер.

Для того щоб виділити всі композитні ребра на скелетизованому бінарному зображенні з відомим положенням ключових точок і вигинів, необхідно виконати наступні кроки:

1. Запустимо алгоритм Лі одночасно з усіх ключових пікселів.

2. Для кожної комбінації стартовою вершини і початкового напрямку руху заведемо стек відвіданих вигинів і кількості кроків на момент їх відвідування.
3. Якщо на черговій ітерації алгоритму хвиля відвідує вершину, відповідну вигину, покладемо в стек мітку цього вигину.
4. Якщо на черговій ітерації алгоритму зустрілися дві хвилі, об'єднаємо два стека цих хвиль, отримавши підсумкову послідовність вигинів в порядку їх відвідування і відстані між двома сусідніми вигинами. Для такої послідовності по початкових позицій хвиль однозначно відновлюється інформація про те, які два ключових пікселя з'єднуються через цю послідовність вигинів.
5. Після закінчення обходу алгоритмом Лі у нас є множина послідовностей вигинів, що характеризують шляху між з'єднаними безпосередньо ключовими пікселями. Розглянемо найкоротші відстані du, v між усіма парами сусідніх вигинів в кожної з послідовностей, а також кількість пікселів pu, v, відвіданих між цими двома вигинами. Позначимо за коефіцієнт викривлення величину відносини pu, v до du, v. Ця величина в достатній мірі дозволяє визначити, наскільки сильно вигнута лінія, що з'єднує два вигини. Порахуємо цю величину для кожного з відрізків між двома з'єднаними вигинами.
6. Перетворимо кожну зі знайдених послідовностей в масив з'єднують ребер. Кожне з таких ребер буде однозначно задаватися мітками вигинів, які ними з'єднуються, напрямних вектором початку і кінця цього ребра, а також коефіцієнтом викривлення. Отриманий масив і є композитним ребром.

Обчислювальна складність такого алгоритму становить O (P), де P – кількість чорних пікселів на скелетизованому бінарному зображенні. Неважко переконатися в тому, що кожен з P таких пікселів в ході роботи алгоритму буде розглянуто один раз.

На рис. 3.9 показані приклади звичайного композитного ребра і композитного ребра, є петлею. Червоним кольором показано одне з можливих

напрямків, в якому могли бути записані відвідані вигини для композитного ребра, синім – аналогічне напрямок для композитного ребра-петлі.

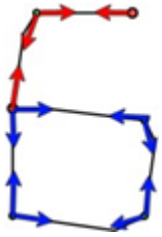


Рисунок 3.9. Виділення композитних ребер

Композитні ребра і ключові точки, які ними з'єднуються, є важливими структурними складовими. Саме на їх основі можна створити структурну модель символу, яка в подальшому буде використовуватися в алгоритмах розпізнавання.

3.6 Алгоритм виділення можливих поділяючих ліній на структурній моделі

Як було зазначено раніше, точками інтересу для вибору місць розташування кордонів між сусідніми символами слова є ключові точки, вигини і центри з'єднуючих ребер. Однак задати розділяє пряму, що проходить через деяку точку інтересу, можна, лише обравши другу точку.

Для вирішення завдання сегментації було зроблено припущення про те, що вибір другої точки можна здійснити так само з множиною точок інтересу.

Процес побудови множини прямих, підмножина яких в підсумку стане розділяють лініями, можна представити у вигляді перебору всіх можливих точок інтересу і перебору в пару кожної з них точки для проведення прямої. Для визначеності можна вважати, що спочатку перебирається нижня з двох точок, а будь-яка розглянута пряма буде згодом задаватися вектором з нижньої точки в верхню. При такому підході кожна пряма розділяє площину структурної моделі на дві півплощини: ліву і праву. Здебільшого елементи структурної моделі, що належать правій півплощині, відносяться до частини слова праворуч від розділової лінії, а що залишилися, відповідно, – до лівої від розділової лінії частини. Варто відзначити, що далі будуть зроблені уточнення, чому не всі точки напівплощин можуть бути віднесені до відповідних частин.

Нескладно помітити, що в результаті роботи описаного раніше перебору буде отримано $O(K^2)$ прямих, які можуть бути розділяють лініями при сегментації, де

K – кількість точок інтересу в структурній моделі. Швидкодію будь-якого алгоритму сегментації, що спирається на дане множина прямих, так чи інакше, буде залежати від розміру цієї множини. Отже, має сенс відсіяти якомога більше свідомо некоректних прямих, які не можуть бути розділовими лініями.

Аналіз наявних структурних моделей рукописних слів дозволив виявити три типи прямих, які можна не розглядати в якості можливих з'єднують ліній:

- горизонтальні або близькі до горизонтальних прямі;
- прямі, які утворені вектором, які перетинають хоча б одне з'єднує ребро рівно в одній точці;
- прямі, які перетинають понад два з'єднуючих ребер.

Далі для обраної прямий слід сформулювати чіткий принцип, за яким слід визначати приналежність кожного з елементів структурної моделі до лівої чи правої від даної прямої частини цієї моделі. Для цього слід спиратися не тільки на геометричні характеристики прямої, а й на місце розташування кінців вектора, яким задається ця пряма. Як вже раніше було сказано, що задає пряму вектор направлений від низу до верху. У такому випадку ми можемо чітко визначити два значення y1 і y2 координати на вертикальній осі Y, між якими знаходиться вектор. Якщо значення y-координати ключової точки або вигину знаходиться між значеннями y1 і y2, то її належність до лівої чи правої від розділової лінії частини визначається тим, зліва чи справа ця точка знаходиться від утворить вектора. Для решти точок можна сформулювати наступний алгоритм визначення приналежності до лівої чи правої частини:

1. Покладемо деякий планарний граф G рівним графу топологічної моделі структурної моделі.
2. Якщо в структурній моделі існує ребро, яке перетинається з відрізком поточної розділяє прямий між y1 і y2, то видалимо з графа G відповідне йому ребро.
3. Пофарбуємо в чорний колір вершини графа G, що відповідають ключовим точкам або вигинів, які знаходяться зліва від утворить поточну розділяє лінію вектора і мають значення y-координати між значеннями y1 і y2.

4. Аналогічно, пофарбуємо в білий колір вершини графа G, що відповідають ключовим точкам або вигинів, які знаходяться праворуч від утворить поточну розділяє лінію вектора і мають значення y-координати між значеннями y1 і y2.
5. Для кожної вершини графа G визначимо компоненту зв'язності, до якої вона відноситься.
6. Розглянемо довільну які раніше не розглянуту компоненту зв'язності графа G.
7. Якщо в розглянутій компоненті зв'язності є розфарбовані вершини тільки одного кольору, то пофарбуємо все вершини цієї компоненти зв'язності в цей колір і перейдемо до кроку 9.
8. Якщо в розглянутій компоненті одночасно є вершини чорного і білого кольору, то помітимо поточну розділяє лінію, як некоректну і завершимо роботу алгоритму.
9. Якщо в графі G є нерозглянуті компоненти зв'язності, перейдемо до кроку 6, в іншому випадку алгоритм закінчений.

Описані критерії дозволяють виділити на вихідної структурної моделі всіх можливих розділяють лінії, які, в свою чергу, утворюють множину ліній-кандидатів L.

3.7 Критерій схожості структурних моделей символів

Тривіальним підходом до вирішення задачі розпізнавання символів є почергове порівняння графічного образу з кожним з еталонних образів, класифікація яких відома, і наступний вибір класу найбільш схожого еталонного образу. У загальному випадку для оцінки ступеня схожості графічному вигляді двох символів на основі структурних моделей, необхідно побудувати структурну модель для кожного з цих символів окремо, після чого дві отримані моделі необхідно порівняти для оцінки їх ступеня схожості. Однак для всіх графічних уявлень бази еталонних зображень можна заздалегідь побудувати структурні моделі і не витрачати обчислювальні потужності на їх побудову при кожному запуску процесу розпізнавання символу.

Оцінка ступеня схожості двох композитних ребер є нетривіальним завданням, враховуючи, що у них можуть відрізнятися кількість складових їх графічних примітивів, типи цих примітивів, їх сумарна довжина, а також місце розташування з'єднуються вигинів і ключових точок.

Однак існує спосіб, за допомогою якого можна визначити площу фігури, укладеної між двома композитними ребрами, не дивлячись на істотні відмінності в їх характеристиках. Цю площу з протилежним знаком можна використовувати як міру схожості між ребрами: чим більше площа, тим більше відрізняються дані ребра.

Крім площі на оцінку ступеня схожості має впливати відстань між відповідними точками композитних ребер – чим далі знаходяться ці точки, тим менше схожі відповідні композитні ребра.

Розглянемо композитне ребро як маршрут, по якому за одну секунду проходить деяких об'єкт. Швидкість об'єкта така, що за цю секунду він пройде повний шлях від одного кінця цього композитного ребра до іншого. Нехай об'єкти почали свій рух одночасно уздовж кожного з ребер. У кожен з можливих моментів часу t проведемо відрізок, що з'єднує положення об'єктів. Сукупність таких відрізків і утворює шукану площу, укладену між двома композитними ребрами.

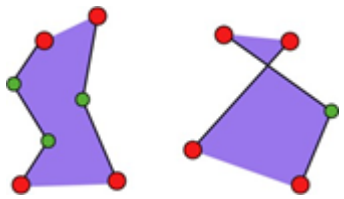


Рисунок 3.10. Площа, яка знаходиться між двома композитними ребрами в якості схожості цих ребер

Навіть якщо композитні ребра мають загальні точки, такий спосіб підрахунку площі між ними може бути використаний для оцінки ступеня схожості.

Розглянемо випадок, коли описаним способом потрібно знайти площу, укладену між двома відрізками. Нехай перший відрізок заданий двома своїми кінцями (x_a, y_a) та (x_b, y_b), аналогічно другий – двома своїми кінцями (x_c, y_c) та (x_d, y_d). Тоді в деякий момент часу t об'єкт на першому відрізку буде знаходитися

в точці (x_1, y_1), на другому – в точці (x_2, y_2). Координати точки (x_1, y_1) можуть бути знайдені за допомогою формули (3.1).

$$\begin{cases} x_1 = x_a + (x_b - x_a) \cdot t \\ y_1 = y_a + (y_b - y_a) \cdot t \end{cases} \quad (3.1)$$

Аналогічно можна визначити координати точки (x2, y2):

$$\begin{cases} x_2 = x_c + (x_d - x_c) \cdot t \\ y_2 = y_c + (y_d - y_c) \cdot t \end{cases} \quad (3.2)$$

Квадрат відстані між ними описується формулою (3.3):

$$d^2 = (x_1 - x_2)^2 + (y_1 - y_2)^2 \quad (3.3)$$

Значення першого доданка формули (3.3) можна представити з використанням формули (3.1):

$$\begin{gathered} (x_1 - x_2)^2 = ((x_a - x_c) + (x_b + x_c - x_a - x_d) \cdot t)^2 = \\ = (A_x + B_x \cdot t)^2 = B_x^2 t^2 + 2 \cdot A_x \cdot B_x \cdot t + A_x^2 \end{gathered} \quad (3.4)$$

Аналогічним чином значення другого доданка :

$$\begin{gathered} (y_1 - y_2)^2 = ((y_a - y_c) + (y_b + y_c - y_a - y_d) \cdot t)^2 = \\ = (A_y + B_y \cdot t)^2 = B_y^2 t^2 + 2 \cdot A_y \cdot B_y \cdot t + A_y^2 \end{gathered} \quad (3.5)$$

Підсумкова формула для обчислення відстані між об'єктами в довільний момент часу, буде мати вигляд:

$$\begin{gathered} d = \sqrt{\left(B_x^2 + B_y^2\right) \cdot t^2 + 2 \cdot \left(A_x B_x + A_y B_y\right) \cdot t + \left(A_x^2 + A_y^2\right)} = \\ = \sqrt{a \cdot t^2 + b \cdot t + c} \end{gathered} \quad (3.6)$$

Використовуючи формулу (3.6), можна знайти площу, укладену між двома розглянутими відрізками:

$$\begin{gathered} S = \int_{t_1}^{t_2} \sqrt{a \cdot t^2 + b \cdot t + c}\, dt = \\ = \frac{1}{8 \cdot a^{3/2}} \cdot \left[\left(2 \cdot \sqrt{a} \cdot (2 \cdot a \cdot t_2 + b) \cdot \sqrt{t_2 \cdot (a \cdot t_2 + b) + c}\right) - \right. \\ -(b^2 - 4 \cdot a \cdot c) \cdot log\left(2 \cdot \sqrt{a} \cdot \sqrt{t_2 \cdot (a \cdot t_2 + b) + c} + 2 \cdot a \cdot t_2 + b\right) - \\ - \left(2 \cdot \sqrt{a} \cdot (2 \cdot a \cdot t_1 + b) \cdot \sqrt{t_1 \cdot (a \cdot t_1 + b) + c}\right) + \\ \left. +(b^2 - 4 \cdot a \cdot c) \cdot log\left(2 \cdot \sqrt{a} \cdot \sqrt{t_1 \cdot (a \cdot t_1 + b) + c} + 2 \cdot a \cdot t_1 + b\right)\right] \end{gathered} \quad (3.7)$$

Формула (3.7) може бути використана для ділянок пари композитних ребер від t1 до t2, на яких немає стиків двох з'єднують ребер, за умови, що обидва з'єднують ребра на цьому відрізку є відрізками. Щоб уникнути надмірно трудомістких обчислень визначених інтегралів для дуг і еліптичних дуг, можна апроксимувати їх послідовністю відрізків. В такому випадку композитне ребро буде представлено все так же послідовністю з'єднуючих ребер, кожне з яких буде відрізком, але кількість їх може бути значно більше.

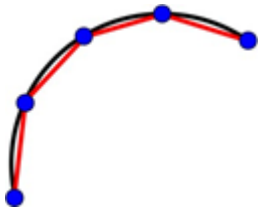


Рисунок 3.11. Аппроксимація дуги відрізками

При такому підході будуть використовуватися не точні значення площі, а наближені. Однак така похибка не надто критична для завдання оцінки ступеня схожості, але розробка і реалізація такого підходу значно простіше, ніж аналогічна реалізація без апроксимації дуг наборами відрізків.

Після того, як для кожної пари композитних ребер порахована площа, яка знаходиться між ними, цю площу можна використовувати як міру відмінності між цими ребрами: чим більше ця площа, тим більше відрізняються ці два ребра. Щоб отримати міру схожості досить помножити отриману площу на мінус одиницю.

Для кожної пари композитних ребер з різних структурних моделей відома їх ступінь схожості. Можна побудувати двочастковий граф G, в якому вершинами будуть композитні ребра вихідних структурних моделей. До першої частці графа G будуть ставитися вершини, які відповідають композитним ребрах з першої структурної моделі, до другої частці – вершини, які відповідають другій моделі.

Для того щоб оцінити ступінь схожості двох структурних моделей, необхідно вершин першої частки зіставити вершини другої частки так, щоб було утворено найбільше можливу кількість пар. З усіх таких розподілів по парам необхідно вибрати розбиття, при якому сума ступенів схожості буде максимальною. Так як ступінь схожості, по суті, є величиною площі з протилежним знаком, то таке

завдання буде рівнозначно задачі пошуку максимального паросполучення мінімальної ваги в дводольному графі, за умови, що ребро такого графа матиме вагу, рівний площі між відповідними композитними ребрами .

Варто відзначити, що частки збудованого графа можуть містити різну кількість вершин. В такому випадку якісь вершини не увійдуть в максимальне паросполучення. Для кожної з таких вершин до загальної ваги знайденого паросполучення необхідно додати подвоєну мінімальну вагу інцидентного йому ребра як «штрафу» за невідповідність кількості композитних ребер у двох моделей.

Завдання знаходження максимального паросполучення мінімальної ваги в дводольному графі зводиться до задачі пошуку максимального потоку найменшої вартості в мережі. Для цього необхідно побудувати новий граф G', який буде мережею з одним джерелом і одним стоком. Щоб отримати такий граф з вихідного двудольного графа досить додати дві вершини. Перша з цих вершин – витік, її необхідно з'єднати орієнтованими ребрами з усіма вершинами з першої частки. Всі такі ребра повинні бути спрямовані від витоку до вершин першої частки. Друга додається вершина – стік, її необхідно з'єднати орієнтованими ребрами з усіма вершинами другий частки. Напрямок таких ребер буде завжди від вершин другої частки до стоку. Всі ребра між частками замінюються аналогічними орієнтованими ребрами з першої частки в другу. Структура отриманого графа G' показана на рис. 3.12:

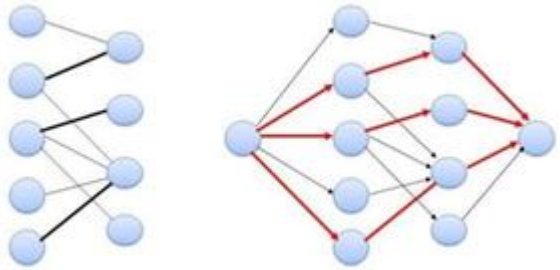


Рисунок 3.12. Схема сітки, отриманої з двудольного графа

Так граф G' є мережею, для якої необхідно визначити величину максимального потоку, для кожного з її ребер визначена величина пропускної здатності. В даному випадку достатньо кожному з ребер присвоїти одиничну

пропускну здатність. Це обумовлено тим, що якщо яке-небудь ребро вже задіяно в поточній парі, то воно більше не має брати участь при утворенні нових пар.

В отриманій мережі необхідно знайти максимальний потік найменшої вартості. Вартість – величина, що характеризує кожне з ребер цієї мережі. Вартість ребра відображає вартість включення цього ребра в знайдений потік. Для побудованої мережі G' вартість всіх ребер з витоку і всіх ребер в стік дорівнює нулю. Вартості ребер, які проводяться між вершинами часткою, відповідають вагам ребер в вихідному дводольному графі G.

Одним з найбільш відомих методів знаходження максимального потоку найменшої вартості є алгоритм збільшуючих шляхів. За своєю суттю цей алгоритм дуже схожий на алгоритм Едмонса-Карпа знаходження максимального потоку.

Для знаходження максимального потоку найменшої вартості в мережі G' необхідно виконати наступні кроки:

1. Попередньо підготуємо граф G', додавши для кожного ребра зворотне ребро з нульовою пропускною спроможністю і вартістю, яка дорівнює вартості вихідного ребра з протилежним знаком.
2. Використовуючи в якості ваг ребер їх вартості, знайдемо найкоротший шлях від джерела до стоку.
3. Якщо не існує жодного шляху від джерела до стоку, алгоритм завершено, знайдений до даної ітерації потік є максимальним потоком мінімальної вартості.
4. Пустимо потік по знайденому маршруту. Для цього розглянемо всі його ребра. Визначимо то, пропускна здатність якого мінімальна, позначимо її за f. Зменшимо пропускну здатність всіх ребер знайденого шляху на f, одночасно збільшуючи пропускну здатність зворотних їм ребер на цю ж величину.
5. Для оновленої мережі знову виконаємо крок 2.

Обчислювальна складність такого алгоритму $O(V^3E)$, де V – кількість вершин в мережі, E – кількість ребер в ній, включаючи додані на першому кроці алгоритму зворотні ребра. Така оцінка характерна, при використанні алгоритму

Дейкстри для пошуку найкоротшого шляху. Швидкодію алгоритму можна поліпшити, використовуючи для реалізації алгоритму Дейкстри бінарну купу.

Так як кількість вершин V в отриманої мережі G', дорівнює $E_1 + E_2 + 2$, то при великій кількості композитних ребер в двох структурних моделях знаходження максимального потоку найменшою вартістю може істотно уповільнити процес розпізнавання символів. На практиці кількість композитних ребер в кожній з моделей рідко перевищує 10, тому алгоритм несуттєво впливає на загальну швидкодію процесу розпізнавання.

Для оцінки ступеня схожості можна використовувати і жадібні алгоритми знаходження максимального парасполучення мінімальної ваги. Так як кожна з вершин першої частки пов'язана з кожною з вершин другої частки, такий алгоритм завжди буде знаходити максимальне парасполучення, проте загальна вага такого парасполучення не завжди буде мінімальним. Проте, паросочетание, знайдене таким способом, можна використовувати в якості наближеної оцінки ступеня схожості.

3.8 Алгоритм вибору підмножини поділу ліній для сегментації структурної моделі

Завдання сегментації окремого слова рукописного тексту була зведена до задачі сегментації структурної моделі, яка, згодом, була зведена до задачі вибору підмножини прямих з множини ліній- кандидатів L, якими буде виконано поділ структурної моделі слова на окремі ділянки, відповідні символам.

Для кожної прямої відомі: її рівняння, початок і кінець вектора, яким утворена ця пряма, а також підмножини ключових точок і вигинів, по кожну зі сторін цієї розділяючої прямої.

Позначимо як G(S) – значення ступеня схожості ділянки структурної моделі, що містить всі пункти інтересів з множини S і з'єднують ребра, обидва кінці яких знаходяться в цій множині, з найбільш схожою еталонної структурною моделлю символу.

Введемо функцію F(S) – максимальне сумарне значення ступенів схожості при поділі множини точок інтересу S деякою підмножиною розділяючих ліній з множини L.

Для того щоб визначити значення функції F(S) для деякого множини точок інтересу S, необхідно розглянути всі розділяючі прямі множини L, які розбивають множину S на дві непусті підмножини. Нехай пряма l_i розбиває множину S на непусті підмножини $S_1^{(i)}$ і $S^{(i)}$. Тоді, розглядаючи всі допустимі прямі l_i, можна обчислити значення F(S), використовуючи рекурентну формулу:

$$F(S) = \max\left(G(S), \underset{i}{max}\, F\left(S_1^{(i)}\right) + F\left(S_2^{(i)}\right) + P_i\right) \tag{3.8}$$

де P_i – величина штрафа за невикористання ділянки структурної моделі в процесі класифікації.

Суть формули полягає у виборі варіанту розбиття множини S таким чином, щоб максимізувати сумарну величину ступеня схожості. Розглядаються всі варіанти розбиття розділяючими прямими множини L і варіант, при якому всі точки множини S відносяться до одного символу, і, отже, подальше розбиття не проводиться.

Обчислення функції F(S) з огляду на рекурентність її аналітичного представлення має експонентну обчислювальну складність. Однак використання кешування вже порахованих значень функції F(S) для певних множин S істотно покращує швидкодію алгоритму обчислення цієї функції. Висока швидкодія обчислень з використанням кешування пояснюється малою кількістю можливих множин S, для яких необхідно обчислити значення функції F(S). У свою чергу, малу кількість можливих множин можна пояснити специфікою способу їх отримання. Кожен раз, за рідкісними винятками, множина точок інтересу, фактично, розбивається на дві підмножини: по ліву і по праву сторону від розділяючої лінії. Таким чином, кількість різних множин на практиці має квадратичну залежність від кількості точок інтересу.

Підхід до обчислення функції F(S) можна охарактеризувати, як застосування методу динамічного програмування. При такому підході стан буде кодуватися деякою хеш-функцією від множини S, а кількість станів, на практиці,

квадратично залежить від кількості елементів вихідної множини, для якої необхідно обчислити значення функції.

Як хеш-функцію можна використовувати легко обчислюваний поліноміальний хеш, який дозволить представити кожну з множин у вигляді відповідного 64-бітного цілого числа. Хешування множини можна виконати, хешируючи упорядкований масив номерів точок інтересу, які входять в дану множину.

3.9 Узагальнення запропонованого алгоритму розпізнавання зображення

Алгоритм виділення структурних складових дозволяє за порівняно невеликий час та за умови обмеженої вибірки еталонних зображень, сформувати якісну структурну модель кожного символа і в подальшому здатний використовуватися для розпізнавання рукописного тексту з досить високою (90-95%) точністю. Однак подібний підхід потребує значних зусиль, затрачених на попередню обробку зображення. На рис. 3.13. зображено загальну блок-схему алгоритму створення структурної моделі символу, здатної як слугувати еталонною моделлю, так і використовуватися для порівняння з іншими моделями (тобто, власне, для розпізнавання символу, з якого сформована модель). Алгоритм складається із наступних основних етапів: введення зображення символу, збільшення контрастності зображення шляхом проведення стандартного алгоритму гістограмної еквалізації, бінаризація зображення (поділ всіх пікселів на чорні та білі в залежності від їх положення відносно певного граничного значення яскравості T), скелетизація зображення (до якої також входить запропонований метод заокруглення гострих закінчень шляхом домальовування чорних пікселів у місцях співпадіння з визначеним переліком піксельних шаблонів), виділення на скелетизованому зображенні ключових елементів та побудова структурної моделі на їх основі.

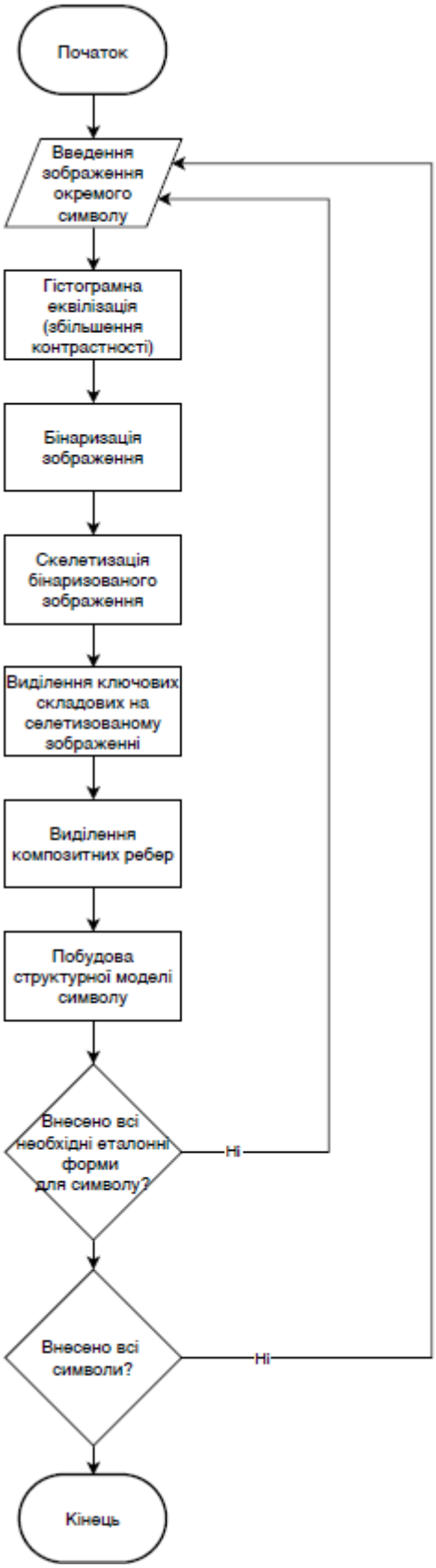


Рисунок 3.13. Загальна блок-схема розробленого алгоритму

Висновки до третього розділу

Розроблено загальну структурну схему системи розпізнавання тексту на зображенні за допомогою формування структурної моделі символу.

Розроблено високопродуктивний алгоритм попередньої обробки вхідного зображення рукописного символу, що включає стадії гістограмної еквалізації, адаптивної бінаризації методом Оцу, скелетизації з використанням методів Зонга-Суня і Ву-Цая.

Запропоновано раціональний підхід до скелетизації бінарного зображення на основі алгоритмів скелетизації Зонга-Суня і Ву-Цая, що має високу швидкодією, в тому числі запропонована продуктивна операція попередньої обробки бінарного зображення з метою усунення плоских закінчень елементів графічного представлення.

Запропоновано алгоритм побудови структурної моделі рукописного символу на основі виділених структурних складових: ключових точок, вигинів, що з'єднуючих і композитних ребер.

Запропоновано алгоритм сегментації рукописного тексту на основі побудови структурної моделі із застосуванням методу динамічного програмування.

Розроблено загальну блок-схему алгоритму, який лежить в основі системи розпізнавання тексту на зображенні.

РОЗДІЛ 4. ПРОГРАМНА РЕАЛІЗАЦІЯ СИСТЕМИ РОЗПІЗНАВАННЯ ТЕКСТУ НА ЗОБРАЖЕННІ ТА ПЕРЕВІРКА ЕФЕКТИВНОСТІ РОЗРОБЛЕНОГО АЛГОРИТМУ

У цьому розділі наведено опис програмної реалізації розроблених алгоритмів. Представлений огляд бібліотек для обробки зображень. Наведено обґрунтування вибору мови програмування і засобів для програмної реалізації алгоритмів. Представлено опис розробленого програмного забезпечення.

4.1 Загальні вимоги до розроблюваного програмного забезпечення системи розпізнавання тексту на зображенні

Сучасні вимоги до якості програмного забезпечення вимагають від розроблюваної системи відповідності ряду характеристик:

- коректне рішення задачі розпізнавання рукописних символів програмної системою;
- коректна реакція програмної системи на некоректні значення вхідних параметрів;
- високу швидкодію програмного забезпечення, що забезпечується відсутністю використання неполіноміальних алгоритмів;
- програмна система повинна використовувати кінцеву кількість оперативної пам'яті і не повинна бути причиною її витоків;
- програма повинна мати модульну архітектуру, а кожен з модулів повинен використовуватися для вирішення окремого завдання.

Розроблювана програмна система для розпізнавання рукописних символів повинна відповідати наступним функціональним вимогам:

1. Можливість виконання попередньої обробки вихідного зображення з метою усунення шуму на оригінальному документі, а також його бінаризації.
2. Виконання операції скелетизації бінарного зображення без втрати важливих структурних елементів графічного зображення символу на основі алгоритмів з поліноміальною обчислювальною складністю.
3. Виділення структурних складових на скелетизованому бінарному зображенні і подальша побудова структурної моделі символу на їх основі.
4. Виконання розпізнавання вхідних зображень з зображеннями рукописних символів з використанням бази структурних моделей еталонних накреслень.
5. Надання користувачеві даних про результати вирішення задачі розпізнавання вхідного зображення.

6. Можливість зміни бази структурних моделей еталонних графічних представлень символів.

4.2 Вибір мови програмування та засобів розробки

При виборі мови програмування для реалізації описаного програмного забезпечення необхідно керуватися рядом вимог:

1. Мова програмування повинна мати високу продуктивність – швидкодія алгоритмів теорії графів і комп'ютерної геометрії може істотно варіюватися залежно від мови.
2. Мова програмування повинна надавати розробнику достатній набір інструментів для реалізації зберігання необхідних даних і оперативного доступу до них: абстрактні типи даних, шаблонізовані контейнери, функтори і т. д.
3. Для мови програмування має існувати множина готових бібліотек і рішень з реалізацією алгоритмів обчислювальної математики, комп'ютерного зору і комп'ютерної геометрії.
4. Мова програмування повинен надавати розробнику можливість самостійно виділяти і вивільняти пам'ять, використовувати бінарні операції, самостійно оперувати з покажчиками на необхідні області пам'яті.

Через вимоги до продуктивності для реалізації алгоритмів побудови і порівняння структурних моделей не підходять сценарні мови програмування, що мають високі функціональні можливості, але невисоку продуктивність.

Мова програмування C++ надає розробнику повний контроль над виділенням і розподілом пам'яті, має високу ступінь гнучкості. Тому ця мова була обрана в якості інструменту розробки програмного забезпечення для розпізнавання рукописних символів.

Для мови програмування C++ існує багато бібліотек з реалізацією методів обчислювальної математики, штучного інтелекту та комп'ютерного зору зокрема. Найбільш відомою з них є бібліотека комп'ютерного зору OpenCV (Open Source Computer Vision Library). У ній представлені засоби для читання,

запису і інтерпретації зображень з різною кількістю каналів і глибиною кольору. Бібліотека OpenCV містить велику кількість класів і функцій для обробки зображень, отримання векторів ознак, а також для класифікації образів.

Бібліотека OpenCV реалізована на мовах C, C ++, також розробляється для Python, Java, Matlab та інших мов і містить наступні модулі:

CXCORE – модуль, який містить базові структури, а також: алгоритми роботи з пам'яттю; алгоритми перетворення типів даних; алгоритми роботи з матрицями; алгоритми роботи з 2D об'єктами.

CV – модуль, призначений для обробки зображень. Містить: алгоритми для обробки і аналізу зображень; алгоритми стеження за об'єктами; алгоритми розпізнавання об'єктів; алгоритми, призначені для калібрування камер.

ML – модуль машинного навчання. Містить: алгоритми, призначені для класифікації образів і аналізу даних.

HighGUI – модуль, призначений для створення призначеного для користувача інтерфейсу. Підтримує: створення вікон, висновок зображень, захоплення відео.

CVCAM – модуль, призначений для захоплення відео з цифрових камер.

Особливий інтерес для системи розпізнавання символів представляють реалізації алгоритмів адаптивної бінаризації і гістограмної еквалізації, а також загальні методи для читання, конвертації, зберігання і запису зображень різних типів в бібліотеці OpenCV.

4.3 Розробка програмної системи

При розробці програмного забезпечення був зроблений вибір на користь модульної архітектури. Такий підхід з використанням декількох окремих модулів, кожен з яких вирішує свою задачу, дозволяє спростити процеси розробки та тестування.

Розроблена програмна система отримала назву «Structure OCR Recognition» і має модульну архітектуру. Кожен з модулів містить функції, які мають схоже функціональне призначення.

4.3.1 Модулі розробленої програмної системи

Взаємозв'язок модулів програмної системи може бути представлений за допомогою діаграми компонентів, представленої на рис. 4.1.

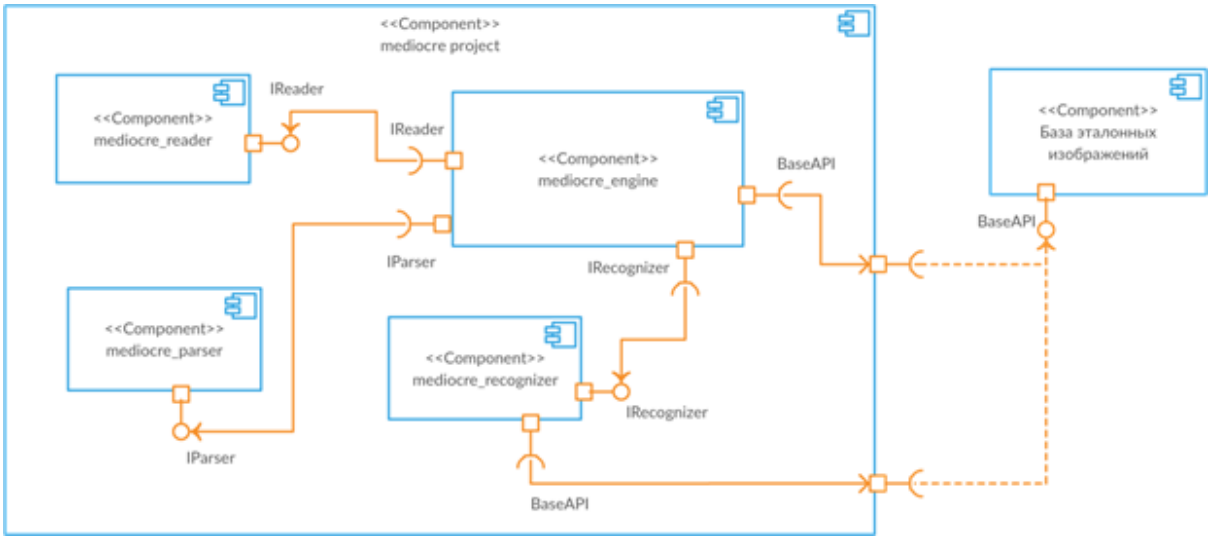


Рисунок 4.1. Діаграма компонентів розробленої програмної системи розпізнавання тексту на зображенні

В таб. 4.1 наведено коротке описання модулів, з яких складається розроблена програма.

Таблиця 4.1

Опис основних модулів програмної системи

Назва модуля	Опис модуля
engine	Основний модуль, приймає значення всіх параметрів від користувача і виконує всі етапи класифікації вхідного зображення рукописного символу. Почергово звертається до інших модулів, які виконують окремі кроки алгоритму
reader	Модуль зчитування вихідного зображення, зміни його розмірів, конвертації в растровий формат. Також виконує стадії первинної обробки зображення: гістограмну еквалізацію і адаптивну бінаризацію. Результат роботи цього модуля може бути використаний в якості вхідних даних для модуля parser

Таблиця 4.1 продовження

Опис основних модулів програмної системи

parser	Модуль виявлення структурних складових і подальшої побудови структурних моделей. Відповідає за попередню скелетизацію растрового бінарного зображення, виділення структурних складових: ключових точок, вигинів, з'єднуючих і композитних ребер, а також побудова структурних моделей. Модуль виконує збереження отриманих моделей на диск у форматі XML.
comparator	Модуль розпізнавання рукописних символів. Оперує набором структурних моделей, відповідних еталонним і вхідному зображенню. Дозволяє виконувати класифікацію описаним способом оцінки ступені схожесті структурних моделей. Результат формує у вигляді XML-файлу – звіту про сканування

Для коректної роботи програми необхідно, щоб робоча станція задовольняла мінімальним вимогам, перерахованим у таблиці 4.2.

Таблиця 4.2

Мінімальні системні вимоги

Назва параметру	**Характеристика**
Операційна система	Windows 7 або новіше
Центральный процесор	З тактовою частотою не нижче 2.2 ГГц підтримує технологію SSE2
Об'єм ОЗУ	Не менше 1 Гб
Місце на жорсткому диску	Не менше 100 Мб

Проект також можна зібрати в операційній системі Linux, адже бібліотеки OpenCV та Eigen є кросплатформеними.

4.3.2. Загальна блок-схема алгоритму програми для розпізнавання зображення за допомогою методу формування структурної моделі символів

Перш за все потрібно вибрати режим роботи: внесення нової еталонної форми, або, власне, порівняння (розпізнавання) тексту. Після цього активується модуль “reader”, в якому реалізовано класи та методи, за допомогою яких відбувається зчитування зображення з вхідного файлу, проводиться його первинна обробка у вигляді збільшення контрасності та бінаризації. З точки зору програми для обох режимів даний етап виглядає однаково. Результат обробки модулем “reader” використовуєтья як вхідні дані до модуля “parser”. Класи та методи цього модуля реалізують алгоритм скелетизації та виділення структурних складових на скелетизованому зображенні. Для режиму розпізнавання також виконується операція сегментації – розділення на окремі символи, придатні до подальшого розпізнавання. І для режиму розпізнавання далі активується модуль “comparator”, в якому відбувається, власне, порівняння відсканованого символу із еталонними формами. Модель еталонного зображення зберігається в окремому XML-файлі, з якого потім зчитується модулем “comparator” для подальшого порівняння із збереженими зображеннями. На рисунку 3.2 можна побачити блок-схему роботи розробленої системи.

4.3.3 Класи для реалізації модуля розпізнавання рукописних символів

У модулі розпізнавання рукописних символів реалізовані всі розроблені в даній роботі алгоритми для виділення структурних складових, побудови структурних моделей, а також оцінки ступеня схожості двох структурних моделей для оцінки ступеня їх схожості. Для їх реалізації використовуються класи і контейнери стандартної бібліотеки шаблонів STL мови програмування C ++, наведені в таблиці 3.3. Крім того, для програмної реалізації модуля були реалізовані класи ocr_image, ocr_comparator і ocr_math, що відповідають за реалізацію логіки процесу розпізнавання.

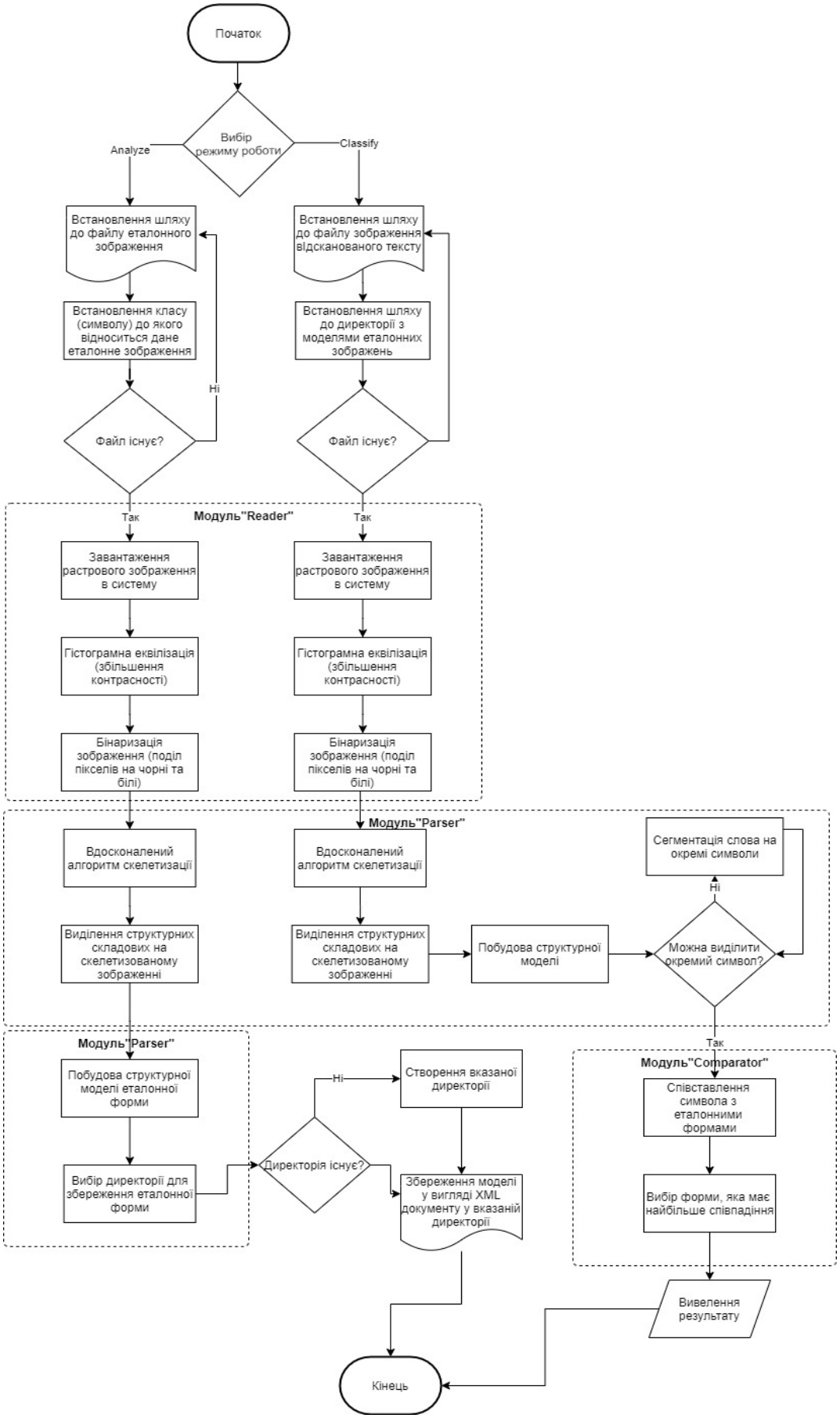


Рисунок 4.2. Блок-схема роботи програми системи розпізнавання тексту на зображенні

Таблиця 4.3

Опис використовуваних класів і контейнерів стандартної бібліотеки шаблонів STL

Назва класу	**Опис**
vector	Послідовний контейнер, який реалізує динамічний масив з підтримкою автоматичної зміни розміру з виділенням і звільненням оперативної пам'яті. У модулі розпізнавання використовується для зберігання колекцій об'єктів, для яких немає необхідності в здійсненні ефективного пошуку, а також швидкого додавання і видалення елементів зі збереженням упорядкованості.
pair	Клас для зберігання пар значень довільних типів. У модулі розпізнавання використовується для зберігання парних геометричних величин, наприклад, координат вигинів і ключових точок.
set	Контейнер унікальних елементів, який реалізує множина. Реалізовано на основі червоно-чорного дерева, що дозволяє додавати і видаляти елементи цього контейнера за логарифмічна час. У модулі розпізнавання використовується для ефективної реалізації алгоритмів виділення ключових точок і вигинів з метою зберігання множин об’єктів, що мають визначені властивості
map	Контейнер, який реалізує словник – аналог асоціативного масиву. Призначений для зберігання відповідностей виду «ключ – значення» без повторюваних ключів. У модулі розпізнавання використовується для бієкції між координатами різних пікселів і групами, до яких вони належать при реалізації алгоритмів виділення структурних складових і побудови структурних моделей.
queue	Контейнер, який реалізує чергу за принципом FIFO (First In, First Out). У модулі розпізнавання використовується для ефективної реалізації алгоритмів скелетизації Зонга-Суня і Ву-Цая.

Реалізовані класи ocr_image, ocr_comparator і ocr_math дозволяють повністю инкапсулировать логіку процесів побудови структурних моделей і їх подальшого зіставлення.

На рис. 4.3. приведена діаграма класів модуля розпізнавання символів.

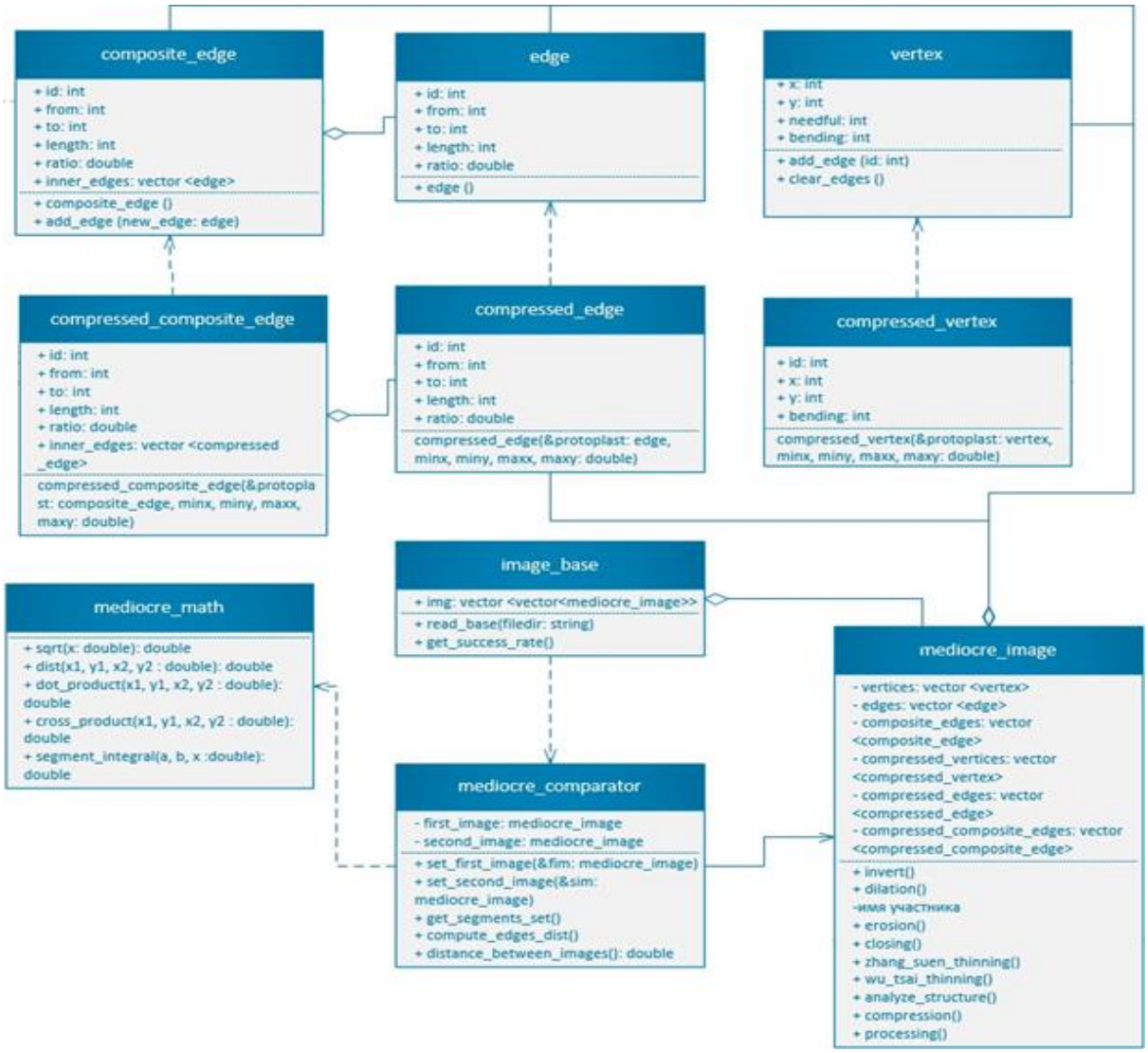


Рисунок 4.3. Діаграма класів модуля розпізнавання символів системи розпізнавання тексту на зображенні

Для створення бази еталонних зображень символів досить зберегти їх структурні моделі у вигляді XML-файлів. При програмної реалізації завантаження бази еталонних зображень досить оголосити контейнер об'єктів класу ocr_image і для кожного з його елементів викликати метод read_from_xml.

4.4 Реалізація збереження структурної моделі символу на жорсткий диск у вигляді XML-файлу

Щоб уникнути необхідності багаторазового обробки одних і тих же зображень символів з метою виділення структурних складових і побудови структурної моделі, при розробці була передбачена можливість збереження отриманої структурної моделі на жорсткий диск у вигляді XML-файла. Також були реалізовані засоби для завантаження збереженої моделі з XML-файла в оперативну пам'ять.

Для реалізації роботи з XML-файлами була використана бібліотека pugixml. Бібліотека позиціонується як легкий простий і продуктивний парсер XML-файлів для додатків, написаних на мові програмування C ++. До її переваг так само можна віднести портіруемость на різні операційні системи, що включають Windows, Linux, MacOS та інші.

Бібліотека pugixml поширюється під роздільною ліцензією MIT і може бути використана при розробці закритого програмного забезпечення за умови, що текст ліцензії буде надано разом з ним.

Сам XML-файл містить теги для опису певних елементів структурної моделі: ключових точок, вигинів, що з'єднують і композитних ребер. Додатково в XML-файлі містяться відомості про вихідне бінарне зображення і результат його попередньої обробки: скелетизації, стискання, визначення ключових пікселів.

4.5 Опис інтерфейсу користувача

Для взаємодії користувача з розробленим програмним додатком передбачений інтерфейс командного рядка та візуальний інтерфейс. Користувач може обрати режим роботи програми: побудова і збереження структурної моделі зображеного символу, порівняння пари структурних моделей або зображень, порівняння зображення або структурної моделі з базою еталонних структурних моделей для вирішення задачі розпізнавання символів.

Оскільки система планується як основа для стартап-проекту (детальніше у розділі 5), в розробленій програмі передбачається активація доступу користувача за допомогою введення коду активації продукту.

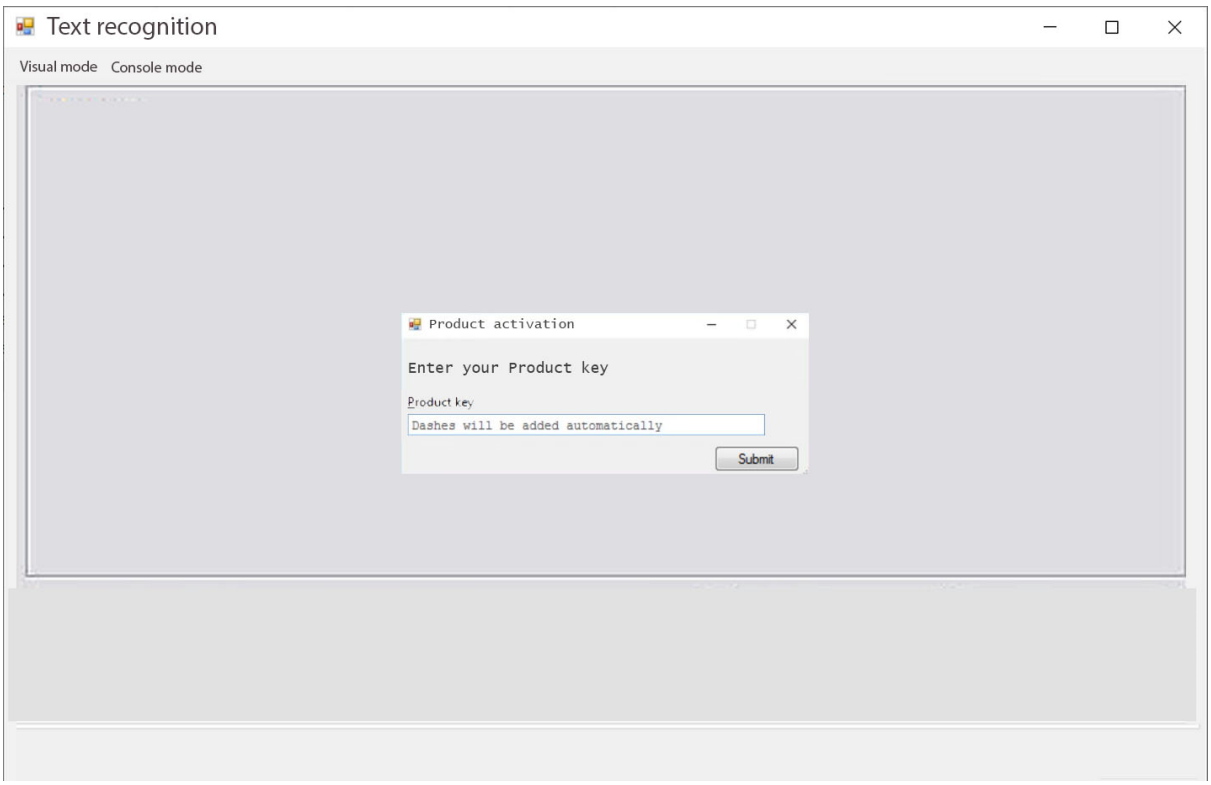


Рисунок 4.4. Активація програмного продукту

Після проходження активації, можна використовувати додаток. В ньому наявні два режими вводу – візуальний та консольний. У консольному режимі використовується найбільш загальний формат команд: [Символ_початку_команди] ім’я_команди [параметр_1 [параметр_2 [...]]], де в квадратних дужках поміщені частини, які не є обов'язковими.

Перелік і опису допустимих параметрів розробленого програмного додатка наведені в таблиці 3.11.

Параметр basedir ігнорується, якщо під час запуску в режимі, що не вимагає використання бази еталонних зображень. У свою чергу, параметри input2 і output2 ігноруються, якщо використовується режим, який не потребує бази еталонних зображень.

Якщо значення output1 або output2 не вказані, то мається на увазі, що збереження структурної моделі в файл не потрібно.

Таблиця 4.4

Допустимі параметри командного рядка

Параметр	**Опис**
Mode	Символьне позначення режиму, в якому слід запустити програмний додаток.
input1	Абсолютний або відносний шлях до файлу з інформацією про перший аналізований символ.
input2	Абсолютний або відносний шлях до файлу з інформацією про другий аналізований символ
output1	Абсолютний або відносний шлях до XML-файлу, в який необхідно зберегти структурну модель першого аналізованого символу.
output2	Абсолютний або відносний шлях до XML-файлу, в який необхідно зберегти структурну модель другого аналізованого символу.
basedir	Абсолютний або відносний шлях до директорії, в якій знаходиться база структурних моделей еталонних зображень.

Залежно від розширень файлів input1 і input2 однозначно визначається, в якому форматі задаються описи вихідних символів: структурними моделями в форматі XML або вихідними зображеннями з графічними представленнями символів.

Для наочності процесів, що відбуваються, користувачу пропонується на розгляд вікно, в якому відтворюється бінаризований символ, результат його скелетизації та кількість виділених точок на зображенні (рис. 4.5)

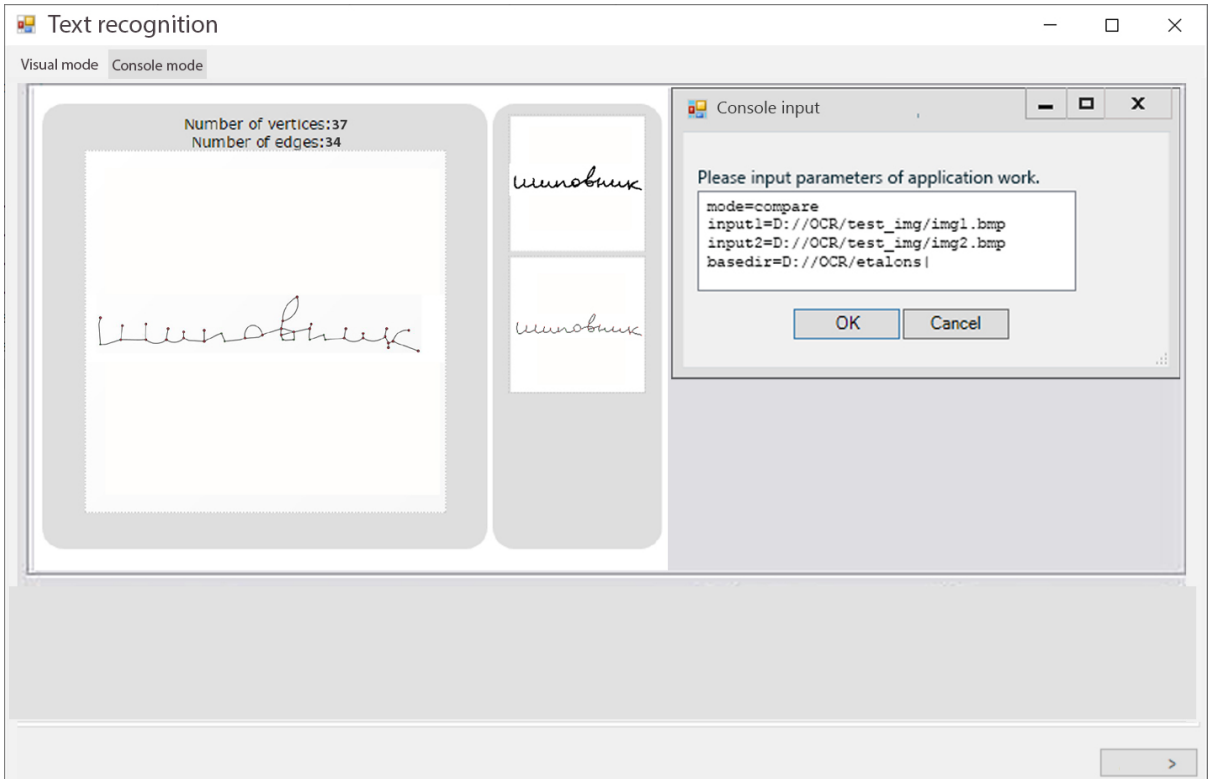


Рисунок 4.4. Консольний режим роботи програми

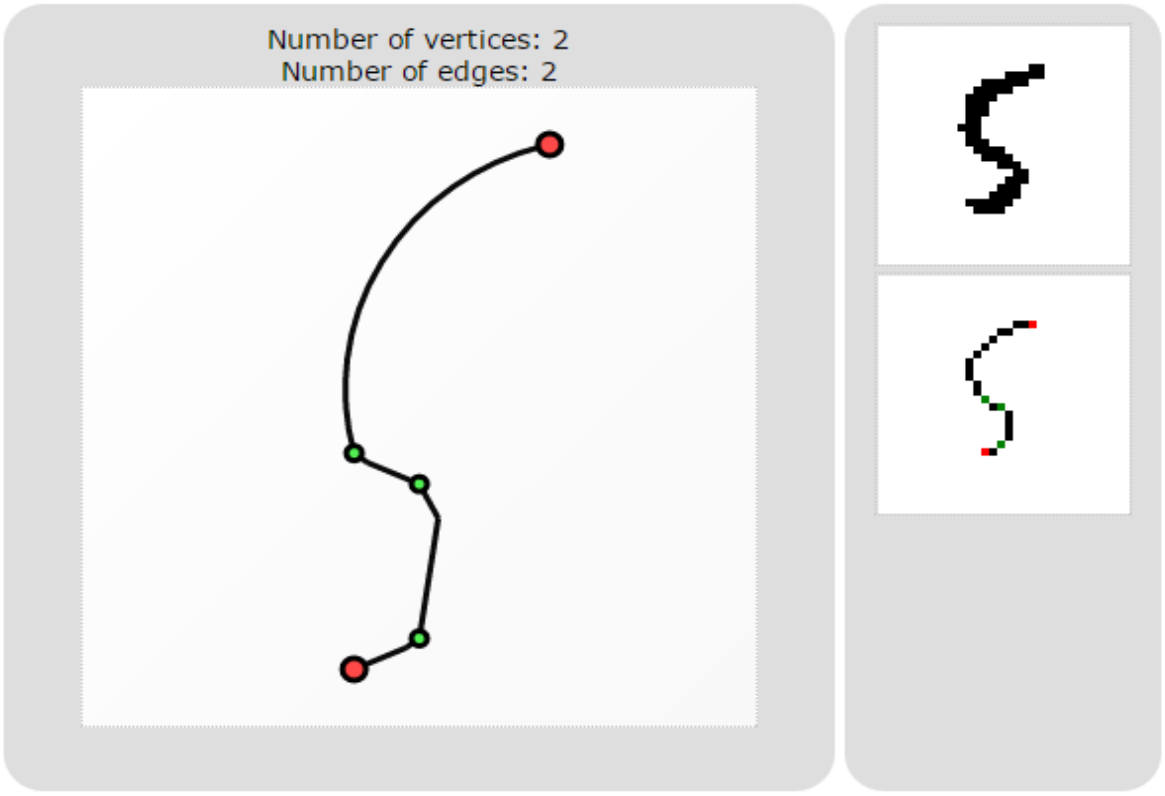


Рисунок 4.5. Вікно візуалізації процесу обробки зображення: бінаризації, скелетизації, виділення ключових структурних елементів

Окрім консольного вводу, можна також скористатись візуальним інтерфейсом. При роботі з візуальним інтерфейсом спочатку потрібно задати директорію, в якій зберігаються еталонні форми.

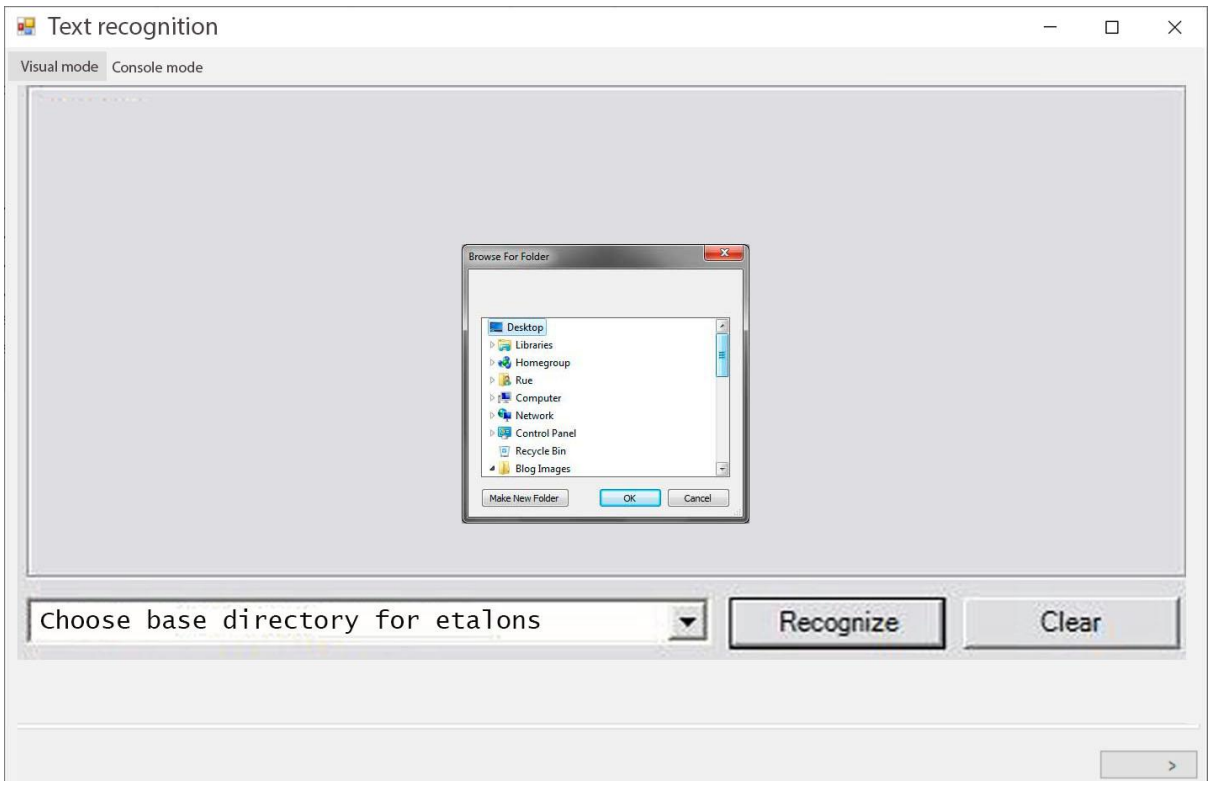


Рисунок 4.6. Вибір папки для зберігання еталонних зображень

На наступному рисунку видно повне вікно візуального інтерфейсу з виведенням результату розпізнавання на екран.

Результат роботи може виводитися як в окремий .txt документ, шлях до якого можна задавати, так і прямо на екран.

Також потрібно вказати директорію, до якої буде збережено файл після запису. Варто зазначити, що для режиму порівняння (mode=compare) збереження файлу не потрібно. Окрім того, для режиму розпізнавання (mode = analyze) замість збереження в окремий файл можна вивести результат одразу на екран в самій програмі.

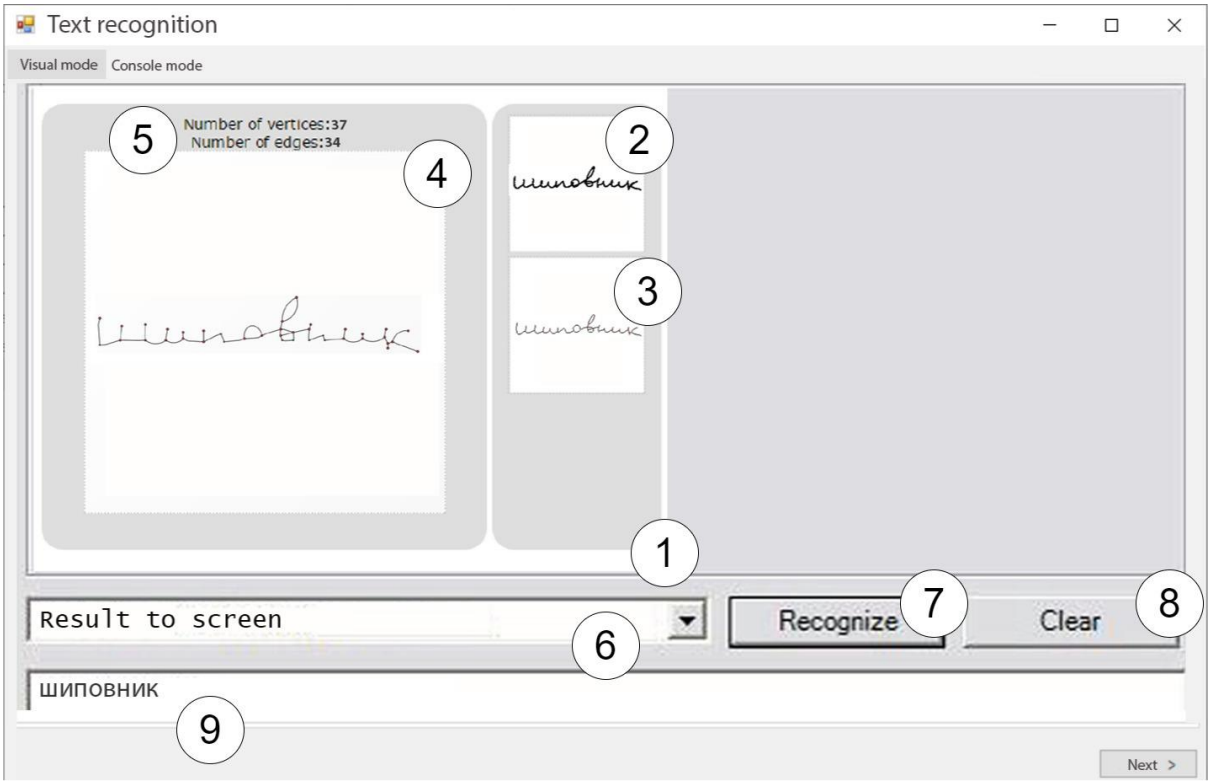


Рисунок 4.7. Візуальний інтерфейс розробленої системи: 1 – Вікно візуалізації процесу обробки. 2 – Результат процесу бінаризації. 3 – Результат процесу скелетизації. 4 – Результат виділення структурних складових. 5 – Лічильник вершин та композитних ребер, виділених у побудованій структурній моделі. 6 – Випадаюче меню вибору шляху виведення результату (передбачена можливість виведення в окремий файл або прямо на екран). 7 – Кнопка запуску процесу розпізнавання. 8 – кнопка очистки вхідних даних. 9 – Вікно виведення результату розпізнавання

Використовуючи консольний режим, директорію для збереження еталонних форм та розташування файлів із зображеннями рукописних символів потрібно вводити вручну. Це може здатися менш зручним, але використання консольного вводу дає суттєво більшу гнучкість. У випадку некорректного вводу в консольному режимі, передбачено обробку помилок та видачу відповідних повідомлень користувачу.

```
This XML file does not appear to have any style information associated with it. The document tree is shown below.

▼<character>
   <rows count="28"/>
   <columns count="28"/>
   <original_pixels
   value="0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
   <pixels
   value="0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
   <original_zoom value="17.0000"/>
   <original_x_shift value="0.3529"/>
   <original_y_shift value="0.6471"/>
 ▼<vertices>
     <vertex id="0" x="0.0000" y="0.7353"/>
     <vertex id="1" x="1.0000" y="0.3824"/>
   </vertices>
 ▼<bendings>
     <bending id="0" x="0.5882" y="0.3824"/>
     <bending id="1" x="0.6471" y="0.5000"/>
     <bending id="2" x="0.9412" y="0.5000"/>
   </bendings>
 ▼<edges>
   ▼<edge id="0" from="0" to="1" length="1.2941" ratio="1.2941">
       &lt;segment x1="0.0000" y1="0.7353" x2="0.5882" y2="0.3824" vx="0.1845" vy="-0.9828" length="0.7647" ratio="1.3000"/&gt;
       &lt;segment x1="0.5882" y1="0.3824" x2="0.6471" y2="0.5000" vx="0.6000" vy="0.8000" length="0.1176" ratio="1.0000"/&gt;
       &lt;segment x1="0.6471" y1="0.5000" x2="0.9412" y2="0.5000" vx="0.8849" vy="0.4657" length="0.2941" ratio="1.0000"/&gt;
       &lt;segment x1="0.9412" y1="0.5000" x2="1.0000" y2="0.3824" vx="0.6000" vy="-0.8000" length="0.1176" ratio="1.0000"/&gt;
     </edge>
   ▼<edge id="1" from="1" to="0" length="1.2941" ratio="1.2941">
       &lt;segment x1="1.0000" y1="0.3824" x2="0.9412" y2="0.5000" vx="-0.2425" vy="0.9701" length="0.1176" ratio="1.0000"/&gt;
       &lt;segment x1="0.9412" y1="0.5000" x2="0.6471" y2="0.5000" vx="-0.8849" vy="0.4657" length="0.2941" ratio="1.0000"/&gt;
       &lt;segment x1="0.6471" y1="0.5000" x2="0.5882" y2="0.3824" vx="-0.2425" vy="-0.9701" length="0.1176" ratio="1.0000"/&gt;
       &lt;segment x1="0.5882" y1="0.3824" x2="0.0000" y2="0.7353" vx="-0.8592" vy="-0.5116" length="0.7647" ratio="1.3000"/&gt;
     </edge>
   </edges>
 </character>
```


Рисунок 4.8. Результат збереження XML-файлу структурної моделі символу

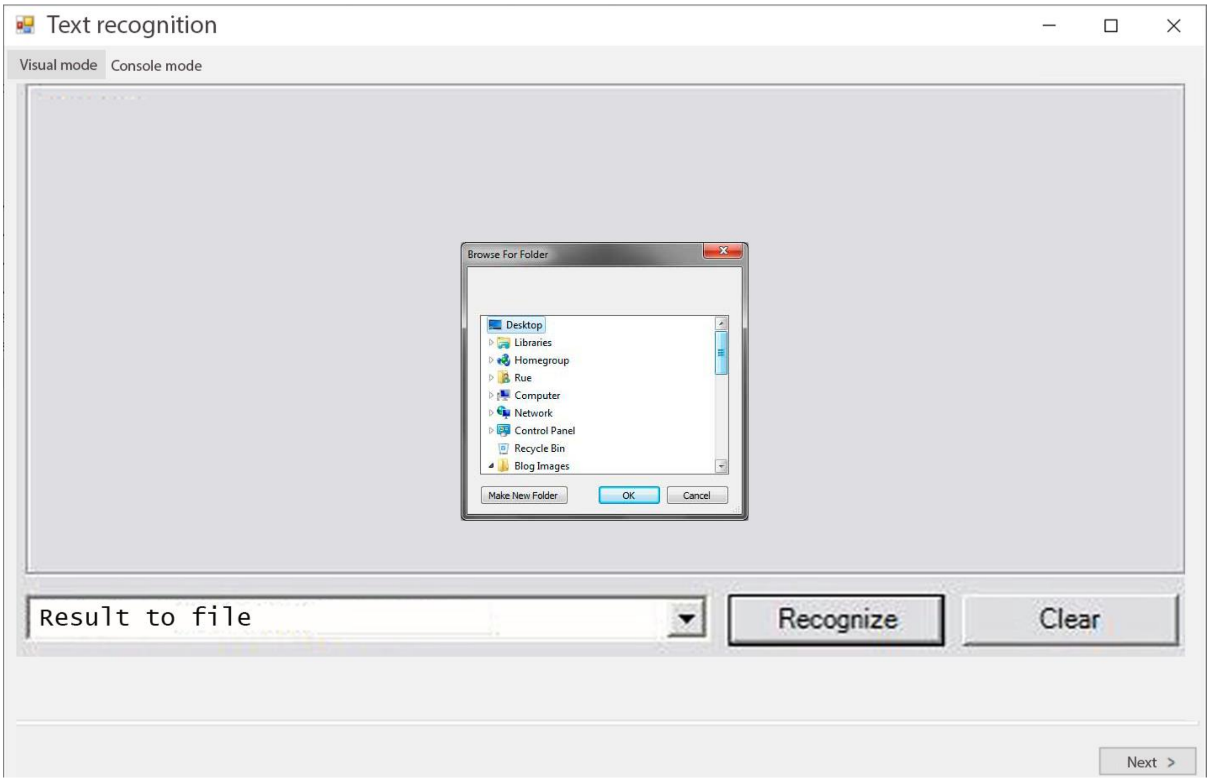


Рисунок 4.9. Збереження файлу розпізнавання

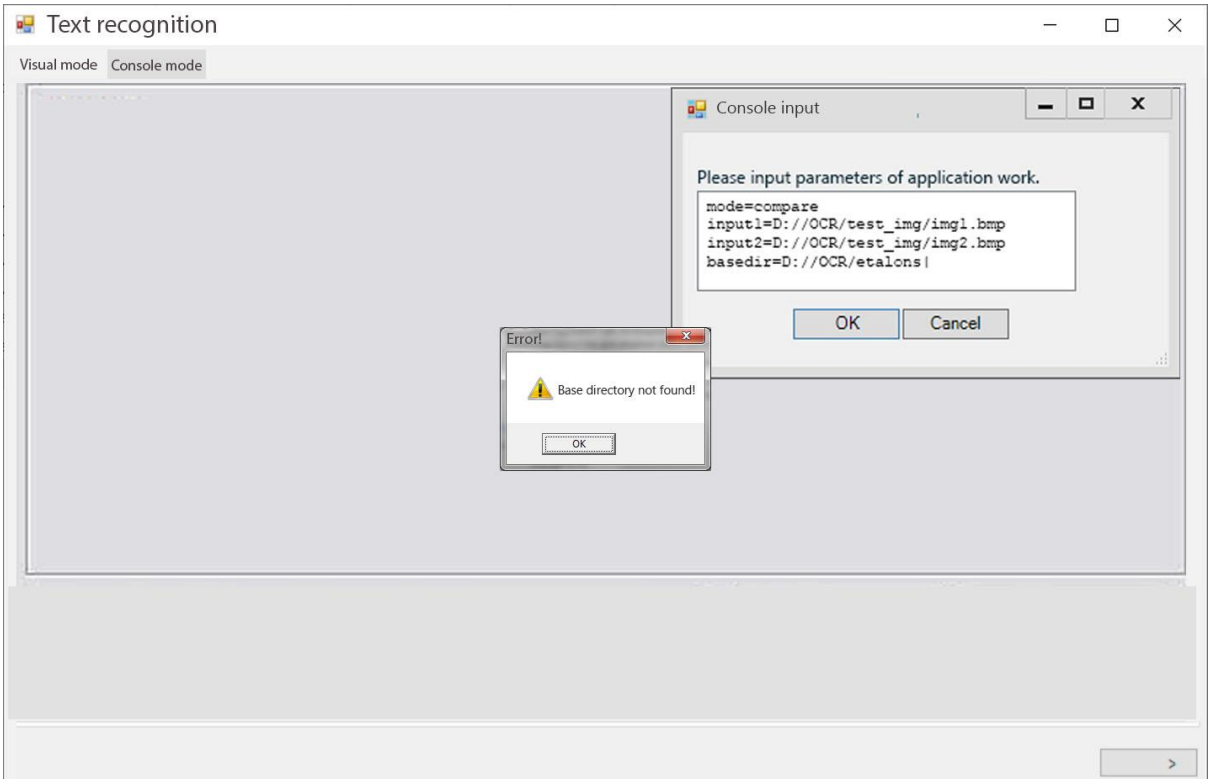


Рисунок 4.10. Обробка помилки при консольному вводі

4.6 Аналіз ефективності побудованої системи розпізнавання символів за допомогою формування структурної моделі

Якщо для вирішення задачі розпізнавання рукописних символів використовується мала кількість еталонних образів, то необхідно забезпечити максимальну різноманітність форм накреслення символів серед зображень, які використовуються в ролі еталонних.

У загальному випадку, це означає, що при виборі еталонних зображень слід вибирати зображення символів, яким відповідають найменш схожі структурні моделі.

Однак варто пам'ятати, що структурні моделі для деяких зображень можуть бути побудовані з істотними помилками. Це може привести до того, що побудована структурна модель не буде відповідати геометричному поданням накресленого на оригінальному документі символу.

На рис. 4.11 наведені досить схожі один на одного зображення символів та візуалізації відповідних їм структурних моделей.

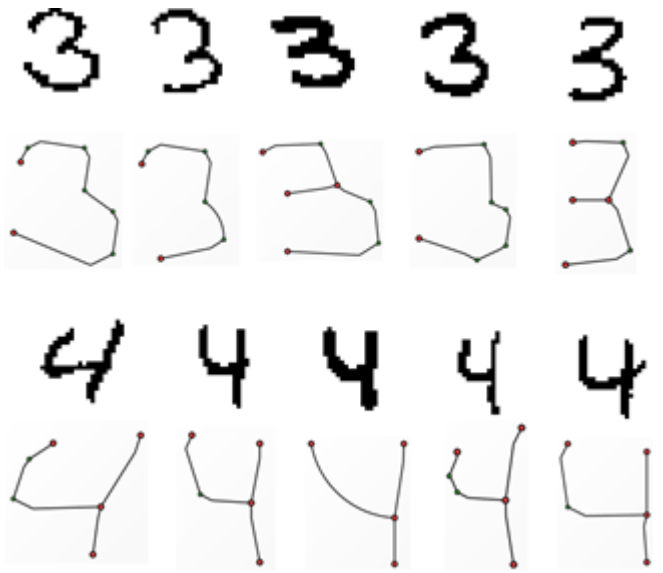


Рисунок 4.11. Зображення рукописних символів і відповідних їм структурних моделей

Як можна помітити, структурні моделі, як правило, відрізняються, але мають досить багато схожих структурних складових. В цілому структурні моделі відповідають геометрії вихідного зображення символу, і погляду на структурну модель без вихідного зображення людині досить, щоб зрозуміти, яким символом вона відповідає.

Структурні моделі двох не схожих між собою накреслень символів одного класу можуть відрізнятися ще більш істотно через недосконалість алгоритмів. Приклад різниці накреслень одного і того ж символу і візуалізації відповідних їм структурних моделей представлені на рис. 4.12.

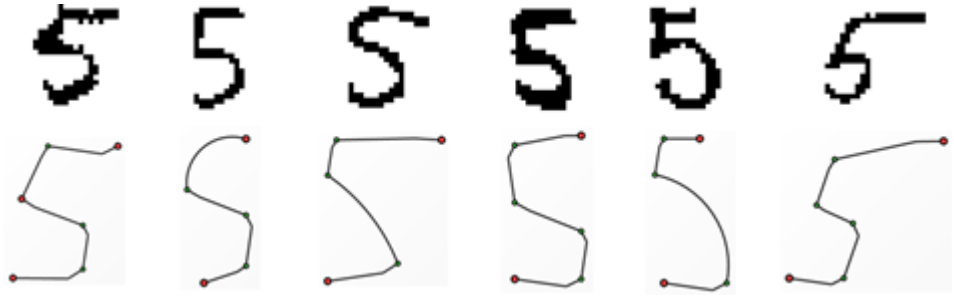


Рисунок 4.12. Зображення з різними зображеннями рукописних символів одного класу і відповідних їм структурних моделей

У випадку з такими зображеннями структурні моделі відрізняються більш істотно: різні пропорції схожих геометричних елементів і різні типи геометричних примітивів, які задають з'єднучі ребра. Як і раніше, структурні моделі досить близько апроксимують початкові представлення символів.

Візуальний аналіз коректності побудованих описаним алгоритмом структурних моделей слів показав, що, в цілому, отримані моделі досить точно описують вихідні графічні представлення слів.

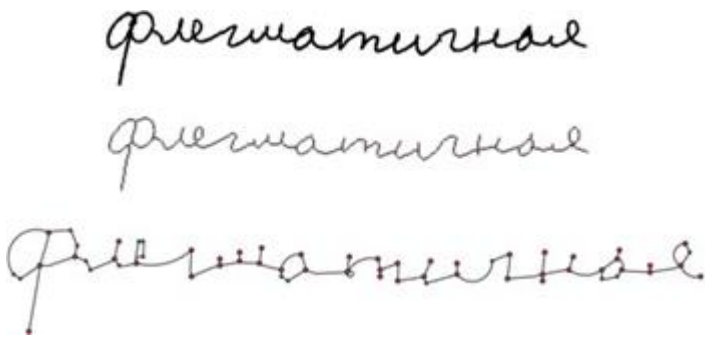


Рисунок 4.13. Приклад накреслення слова, результату його скелетизації і структурної моделі

Для оцінки якості розпізнавання символів в умовах малої навчальної вибірки, запропонований алгоритм був порівняний з аналогами, здатними функціонувати в умовах малої кількості еталонних зображень:

- метод перетинів для растрових зображень (IM);
- алгоритм на основі методу опорних векторів (SVM);
- метод на основі імовірнісної нейронної мережі (PNN);
- гістограмного метод з in-out і out-in профілями (HMP).

Для навчання або настройки в кожному з експериментів використовувалося різна кількість еталонних зображень E. Експерименти проводилися для непарних значень E від 3 до 11. Зображення для перевірки було взято із бази MNIST.

Результати експериментів для набору MNIST наведені в таб. 4.4.

Як можна помітити з наведених результатів експериментів, алгоритм на основі пошуку максимального паросполучення мінімальної ваги частіше вірно класифікує зображення рукописних символів при заданих значеннях кількості еталонних зображень.

При значенні E = 5 перевага такого методу найістотніша.

Важливо так само відзначити, що при збільшенні кількості еталонних зображень до декількох тисяч якість розпізнавання запропонованого методу поліпшується несуттєво, в той час як їх швидкодія істотно погіршується.

Таблиця 4.4

Відсоток вірно розпізнаних символів набору MNIST кожним з порівнюваних алгоритмів

E	SMM	IM	SVM	PNN	HMP
3	93,2	87,4	83,8	70,9	74,1
5	95,1	88,7	85,5	73,2	74,3
7	95,1	88,9	87,0	74,2	74,8
9	95,2	90,4	88,2	74,7	75,0
11	95,2	90,4	88,5	75,3	75,1
13	95,3	74,1	74,1	74,1	74,1
15	95,3	74,3	74,3	74,3	74,3

Висновки до четвертого розділу

Сформульовані основні вимоги до засобів розробки програмних додатків: мова програмування та підключені бібліотеки. На основі даних вимог зроблено вибір засобів для програмної реалізації.

Сформовано загальну блок-схему роботи розробленої програми.

Викладено опис основних класів, що використовуються в модулях розпізнавання символів. Описані основні атрибути, змінні та методи цих класів. Сформовано UML-діаграму класів та компонентів.

Представлений запропонований формат XML-файлу для збереження інформації про структурні моделі символів на диску.

Розроблено програмну систему із використанням обраних програмних засобів. Представлено детальний перелік елементів системи та опис користувацького інтерфейсу. Надано зображення основних етапів використання програми.

Проведено порівняння ефективності алгоритму розробленої системи із аналогами, що можуть працювати в умовах малої вибірки. Встановлено, що в подібних умовах запропонована система значно більш ефективна для вирішення завдань розпізнавання рукописних символів.

РОЗДІЛ 5. МАРКЕТИНГОВИЙ АНАЛІЗ СТАРТАП-ПРОЕКТУ

Базовою стратегією стартапу є задоволення спільних інтересів шляхом формування прийнятної пропозиції з обопільною корисністю. Однак, створення та ринкове впровадження стартап-проектів відзначається підвищеною мірою ризику, ринково успішними стає лише невелика частка.

У даному розділі наведена розробка проектної пропозиції для проекту, що було створено у рамках магістерської дисертації. Очевидно, що результати даної дисертаційної роботи є важливими з соціальної та економічної точок зору. Дана робота є інноваційною та потрібною суспільству. Основна суть роботи полягає у розробці системи розпізнавання тексту на зображенні.

5.1. Опис ідеї проекту

З кожним днем все актуальнішою стає задача переведення багаточисельних рукописних текстів, таких як старі архіви, конспекти, та будь-які інші записи, в електронний, зручний для зберігання та подальшого багаторазового використання формат. Одночасно з тим, звичайні підходи до вирішення цієї задачі, такі як ШНМ часто потребують значних зусиль для навчання та настройки, зокрема вимагають наявності обширної навчаючої вибірки, що далеко не завжди зручно та можливо досягти. Таким чином система розпізнавання тексту на зображенні із використанням малої навчаючої вибірки, високою точністю та швидкодією матиме перспективу на ринку.

Таблиця 5.1

Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Проблема полягає у тому, що на даний момент не існує єдиної стабільно працюючої нейронної мережі (або іншого засобу), який би дозволяв чітко та стабільно розпізнавати рукописний текст різних шрифтів та почерків. Стартап націлений на вирішення цієї проблеми.	Ряд завдань, в яких кількість спочатку відомих накреслень символів вкрай невелика. Прикладами таких завдань є: розпізнавання заповнених нетиповим почерком бланків атестації, виділення текстової інформації на наявних в єдиному екземплярі історичних документах, тощо	Заощадження часу; гарантована правильність розрахунків; наявність супровідних пояснювальних даних.

Проведемо аналіз потенційних техніко-економічних переваг ідеї.

Таблиця 5.2

Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№ н/п	Техніко-Економічні характеристики ідеї	(потенційні) товари/концепції конкурентів			W (слабка сторона)	N (нейтральна сторона	S (сильна сторона)
		Запропонована система	Abby FineReader	Tesseract OCR			
1	Вартість ліцензії	низька	висока	відсутня	-	+	-
2	Складність налаштування для використання	низька	низька	висока	-	-	+
3	Точність роботи алгоритму для рукописного тексту	Висока	Низька	Середня	-	-	+
4	Швидкість виконання	Висока	Висока	Середня	-	-	+
5	Крос-платформена робота	Так, можна запускати на різних ОС	Так	Так	-	-	+

5.2. Технологічний аудит ідеї проекту

Визначення технологічної здійсненності ідеї проекту передбачає аналіз таких складових (таб. 5.3):

- за якою технологією буде виготовлено товар згідно ідеї проекту?
- чи існують такі технології, чи їх потрібно розробити/доробити?
- чи доступні такі технології авторам проекту?

В таблиці 5.3 проводиться аудит технології, за допомогою якої можна реалізувати ідею проекту.

Таблиця 5.3

Технологічна здійсненність проекту

№ п/п	Ідея проекту	Технологіїїї реалізації	Наявність технологій	Доступність технонологій
1.	Створення системи розпізнавання тексту на зображенні, ефективної при малій навчаючій вибірці	Visual Studio, Windows Forms, C++, OpenCV,	Технологія наявна	Технологія доступна, проблем розробки не передбачається

Отже, можна зробити висновок, що головними перевагами для користувача є точність, легкість налаштування та кросс-платформеність системи.

5.3. Аналіз ринкових можливостей запуску стартапу

Визначення ринкових можливостей, які можна використати під час ринкового впровадження проекту, та ринкових загроз, що можуть перешкодити реалізації проекту, дозволяють спланувати напрями розвитку проекту із урахуванням стану ринкового середовища, потреб потенційних клієнтів та пропозицій проектів конкурентів.

Рентабельність – поняття, що характеризує економічну ефективність виробництва, за якої за рахунок грошової виручки від реалізації продукції (робіт, послуг) повністю відшкодовує витрати на її виробництво й одержується прибуток як головне джерело розширеного відтворення.

Проведемо аналіз попиту: наявність попиту, обсяг, динаміка розвитку ринку (таб. 5.4).

Таблиця 5.4

Попередня характеристика потенційного ринку стартап-проекту

№ п/п	*Показники стану ринку (найменування)*	*Характеристика*
1	Кількість головних гравців, од	1
2	Загальний обсяг продаж, грн/ум.од	40000
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Немає, оскільки цей ринок не контролюється якимись органами держави
6	Середня норма рентабельності в галузі (або по ринку), %	20% на рік

Провівши аналіз попиту, можна зробити висновок, що коефіцієнт рентабельності достатньо високий. Далі визначаємо потенційні групи клієнтів, їх характеристики, та формуємо орієнтовний перелік вимог до товару для кожної групи (таб. 5.5).

Таблиця 5.5

Характеристика потенційних клієнтів стартап-проекту

№ п/п	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Ефективне розпізнавання рукописного тексту на зображенні	Держслужбовці, пов'язані з архівами, перевіряючі атестаційних робіт	Перевага надається новим алгоритмам з покращеними характеристиками	Низька ціна та висока точність кінцевого продукту
2	Індивідуальне налаштування системи розпізнавання з мінімальними зусиллями	Необхідність сканувати текст на мові із специфічними символами (не латиниця)	Перевага надається системам, здатним працювати з невеликою навчальною вибіркою	Низька ціна та висока точність кінцевого продукту

Ринкові можливості – це сприятливі обставини, які підприємство може використовувати для отримання переваг. Як приклад ринкових можливостей можна привести погіршення позицій конкурентів, різке зростання попиту, появу нових технологій виробництва продукції, зростання рівня доходів населення і т. п. Слід зазначити, що можливостями з погляду SWOT-аналізу є не всі можливості, які існують на ринку, а тільки ті, які можна використовуватиАналіз ринкового середовища.

Таблиця 5.6

Фактори загроз

№ п/п	*Фактор*	*Зміст загрози*	*Можлива реакція компанії*
1	Вихід конкурентів на ринок	Після виходу продукту на ринок можлива поява конкурентів	Нові інновації для приваблення нових клієнтів
2	Криза	Зменшення обсягу продаж	Спрощення роботи програми

Таблиця 5.7

Фактори можливостей

№ п/п	*Фактор*	*Зміст можливості*	*Можлива реакція компанії*
1	Нові сфери використання	Крім використання копірайтерськими компаніями можливе й інше застосування	Націлення на нові сфери використання для приваблення більшої кількості клієнтів
2	Інтеграція на нові платформи	Можливість використовувати не тільки на ОС Windows, а й на інших ОС та на мобільних телефонах	Приваблення нових розробників для інтеграції з новою платформою, розростання компанії

Разом з розширенням ринку та плином часу фактори можливостей будуть збільшуватись, разом з тим будуть збільшуватись і загрози, тому необхідно швидко реагувати за можливості і загрози.

Проведемо аналіз пропозиції (таб. 5.8) та визначимо загальні риси конкуренції на ринку:

Таблиця 5.8

Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	*В чому проявляється дана характеристика*
1. Чиста конкуренція	Окремі покупці і продавці не можуть впливати на ціну.
2. Національна конкуренція	Між компаніями однієї країни
3. Внутрішньогалузева конкуренція	Конкурентна боротьба між підприємствами в межах однієї галузі.
4. Товарно-видова конкуренція	Конкуренція між товарами одного виду.
5. Нецінова конкуренція	Додавання нового якісного функціоналу
6. Марочна конкуренція	Конкурентні компанії пропонують подібний продукт.

Провівши ступеневий аналіз на ринку, можна зробити висновок, що на ринку достатньо конкурентоспроможних систем, проте як вже було зазначено

вище, використання алгоритму, здатного досягти високої точності при мінімальному об’ємі еталонної вибірки дає змогу зайняти малозаповнену нішу, при цьому не збільшуючи кінцеву вартість продукту.

Далі проведемо більш детальний аналіз умов конкуренції в галузі за моделлю п’яти сил М. Портера (таб. 5.9)

Таблиця 5.9

Аналіз конкуренції в галузі за М. Портером

Складові аналізу	*Прямі конкуренти в галузі*	*Потенційні конкуренти*	*Постачальники*	*Клієнти*	*Товари-замінники*
	PDF OCR	*Розмір капіталовкладень*	*Якість розпізнаного тексту*	*Своживачі повинні весь час користуватися функціями саппорту*	*Замінники є безкоштовними*
Висновки:	Інтенсивність не є великою, оскільки функціональні можливості конкурента значно гірші	Можливості входу є, оскільки за допомогою служби саппорт можна значно поліпшити процес інтеграції. Строки виходу на ринок: одразу після релізу готового продукту	Не диктують	Не диктують	Серед обмежень можна відмітити небезкоштовність товару

Провівши аналіз конкуренції в галузі за М. Портером видно, що є достатня кількість прямих конкурентів, що гарно зарекомендували себе на ринку, тому необхідно постійно слідкувати за якістю та точністю роботи своєї системи.

На основі аналізу конкуренції в галузі, що наведено в таблиці 5.9, а також із урахуванням характеристик ідеї проекту, які були розглянуті в таблиці 5.2, вимог споживачів до товару (таб. 5.5.) та факторів маркетингового середовища (таб. 5.6, 5.7) визначимо та обґрунтуємо перелік факторів конкурентоспроможності. Аналіз конкурентоспроможності представлено в таблиці 5.10.

Таблиця 5.10

Обґрунтування факторів конкурентоспроможності

№ п/п	*Фактор конкурентоспроможності*	*Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)*
1	Функціональність	У конкурента дуже дрібна функціональність, що не може конкурувати з функціональністю даного продукту
2	Швидкість роботи	У конкурента не оптимізовані методи та алгоритми розробки, що робить парсинг дуже повільним
3	Навантаженість на систему	У конкурента програма працює в режимі онлайн, що потребує затрат на інтернет трафік
4	Об’єм еталонної вибірки	Методи розпізнавання конкурента потребують великих навчаючих виборок для стартового налаштування

Як можна побачити з таблиці було обґрунтовано основні фактори конкурентоспроможності, основними з яких стали: функціональність, швидкодія, продуктивність, та простота налаштування.

Проведемо порівняльний аналіз сильних та слабих сторін факторів конкурентоспроможності (таб. 5.11).

Таблиця 5.11

Порівняльний аналіз сильних та слабких сторін розробленої системи

№ п/п	*Фактор конкурентоспроможності*	*Бали 1-20*	*Рейтинг товарів-конкурентів у порівнянні з … (назва підприємства)*						
			–3	*–2*	*–1*	*0*	*+1*	*+2*	*+3*
1	Функціональність	18					1	1	2
2	Швидкість обробки	20					1	1	1
3	Навантаженість на систему	12					1	2	3
4	Максимальний об’єм для обробки	10					1	2	3

З таблиць 4.10 та 4.11 бачимо, що фактори конкурентоспроможності суттєві та мають великий позитивний внесок при впроваджені на ринок пристрою «RadioInstant». Основною перевагою даного пристрою є висока точність алгоритму та низька ціна.

Далі проведемо SWOT-аналіз стартап-проекту, що наведено в таблиці 5.12.

Таблиця 5.12

SWOT- аналіз стартап-проекту

Сильні	Слабкі
Маркетинг	
Можливість застосувати поширені типи реклами та маркетингу	Імідж відсутній, впізнаваність мінімальна, необхідне докладання значних зусиль та ресурсів
Виробництво	
Нескладне, необхідний лише безкоштовне доступне середовище Visual Studio	ПЗ знаходиться у відкритому доступі, що полегшує вихід на ринок конкурентів
Персонал	
Досить просто знайти персонал ІТ-галузі	Невисока кількість спеціалістів з робототехніки
Дослідження та розробки	
Технології відомі, додаткові дослідження не затребувані	Залежність від одного виду середовища розробки Visual Studio та умов його використання
Фінанси	
Для запуску виробництва потрібна невелика кількість фінансових ресурсів	Необхідність сплачувати податки може стати проблемою на початковому етапі низьких доходів
Можливості	Загрози
Вихід на закордонні ринки	Витіснення конкурентами
Розширення переліку створюваних додатків та їх функції	Відсутність достатнього попиту
Підвищення впізнаваності та створення бренду	Проблеми з наявність ресурсів для маркетингових компаній
Створення безкоштовної версії для обмеженої кількості клієнтів або обмеженої версії для всіх клієнтів для запуску саморозповсюджувальної реклами	Погіршення економічного та політичного станів в країні
Укласти партнерський договір з виробниками роботів	Зниження купівельної спроможності

Для успішності впровадження стартап-проекту на ринку необхідно враховувати появу ризиків та слабких сторін проекту. Наприклад, для покращення репутації пристрою необхідно враховувати побажання користувачів та постійно оновлювати алгоритми роботи.

На основі SWOT-аналізу розробимо альтернативи ринкової поведінки для виведення стартап-проекту на ринок та орієнтовний оптимальний час їх ринкової реалізації з огляду на потенційні проекти конкурентів, що можуть бути виведені на ринок.

Визначені альтернативи аналізуються з точки зору строків та ймовірності отримання ресурсів

Таблиця 5.13

Альтернативи ринкового впровадження стартап-проекту

№ п/п	*Альтернатива (орієнтовний комплекс заходів) ринкової поведінки*	*Ймовірність отримання ресурсів*	*Строки реалізації*
	Створення максимально оптимізованих методів розпізнавання	Висока ймовірність (80%)	Великі строки реалізації (2 місяці)
	Потужна реклама продукту, яка зацікавлює клієнтів	Мала ймовірність (30%)	Середні строки реалізації (1 місяць)
	Пропонування нових сфер використання продукту	Висока ймовірність (90%)	Середня строки реалізації (1 місяць)

5.4. Розроблення ринкової стратегії проекту

Розроблення ринкової стратегії першим кроком передбачає визначення стратегії охоплення ринку: опис цільових груп потенційних споживачів.

Таблиця 5.14

Вибір цільових груп потенційних споживачів

№ п/п	*Опис профілю цільової групи потенційних клієнтів*	*Готовність споживачів сприйняти продукт*	*Орієнтовний попит в межах цільової групи (сегменту)*	*Інтенсивність конкуренції в сегменті*	*Простота входу у сегмент*
	Компанії, що займаються копірайтерством	Продукт є поліпшенням існуючих варіантів – а тому готовність прийняття має бути високою	Через малу функціональність альтернатив попит на цей продукт має бути високим	Даний продукт поєднує в собі більше функцій, ніж програми конкурентів	Не висока
Які цільові групи обрано: Копірайтерські компанії					

Таблиця 5.15

Визначення базової стратегії розвитку

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку**
	Випустити першу версію продукту з усім запланованим функціоналом	Пропозиція найбільш оптимізованих методів обробки інформації серед існуючих (ставка на універсальність)	Співвідношення простоти та функціональності	Стратегія диференціації

Таблиця 5.16

Визначення базової стратегії конкурентної поведінки

№ п/п	*Обрана альтернатива розвитку проекту*	*Стратегія охоплення ринку*	*Ключові конкурентоспроможні позиції відповідно до обраної альтернативи*	*Базова стратегія розвитку**
	Випустити першу версію продукту з усім запланованим функціоналом	Пропозиція найбільш оптимізованих методів обробки інформації серед існуючих (ставка на універсальність)	Співвідношення простоти та функціональності	Стратегія диференціації

Таблиця 5.17

Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції стартап- проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Правильність розрахунків	Стратегія диференціації	Коректність	Висока якість
2	Простий інтерфейс	Стратегія диференціації	Зрозумілість	Простота
3	Наявність підказок та супровідного матеріалу	Стратегія диференціації	Дружність інтерфейсу	Супровід

Визначена базова стратегія розвитку проекту – стратегія спеціалізації, оскільки ця стратегія передбачає концентрацію на потребах одного цільового сегменту, без прагнення охопити увесь ринок. Мета тут полягає в задоволенні потреб вибраного цільового сегменту краще, ніж конкуренти. Така стратегія

може спиратися як на диференціацію, так і на лідерство по витратах, або і на те, і на інше, але тільки у рамках цільового сегменту. Тоді, точність роботи алгоритму дає можливість встановлювати вищу ціну на продукцію, так як споживачі готові їх сприйняти.

5.5. Розроблення маркетингової програми стартап-проекту

Спершу формується маркетингова концепція товару для споживача (таб. 5.18).

Таблиця 5.18

Визначення ключових переваг концепції потенційного товару

№ п/п	*Потреба*	*Вигода, яку пропонує товар*	*Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити*
1	Універсальність, швидкість обробки	Висока швидкість та функціональність	Функціональні можливості, оптимізація методів розпізнавання зображень/рукописних текстів

Наступною відбувається розробка трьохрівневої маркетингової моделі товару: уточнення ідеї продукту, його фізичні складові (таб. 5.19).

Таблиця 5.19

Опис трьох рівнів моделі товару

<table>
<tr><td>Рівні товару</td><td colspan="3">Сутність та складові</td></tr>
<tr><td>I. Товар за задумом</td><td colspan="3">Програма повинна приймати файли без обмеження на їх розмір та оброблювати файли за малий проміжок часу. Отримані електроні файли за змістом мають повністю співпадати з рукописними файлами/зображеннями</td></tr>
<tr><td rowspan="6">II. Товар у реальному виконанні</td><td>Властивості/характеристики</td><td>М/Нм</td><td>Вр/Тх /Тл/Е/Ор</td></tr>
<tr><td>1. Швидкість розпізнавання
2. Розмір файлів
3. Навантаженність
4. Функціональність</td><td>М
М
М
М</td><td>Тх
Тх
Е
Тх</td></tr>
<tr><td colspan="3">Якість: гарантована висока швидкість розпізнавання та якість (не має норм та стандартів)</td></tr>
<tr><td colspan="3">Пакування: Maven репозиторій</td></tr>
<tr><td colspan="3">Марка: MultiConvert</td></tr>
<tr><td colspan="3">Існує система зворотнього зв'язку з користувача з розробником</td></tr>
<tr><td>III. Товар із підкріпленням</td><td colspan="3">Пропонується періодичне оновлення програми.</td></tr>
<tr><td colspan="4">За рахунок чого потенційний товар буде захищено від копіювання: за рахунок універсальності та новітніх ідей для створення алгоритмів розпізнавання</td></tr>
</table>

Наступним етапом є визначення рівня цін на потенційний товар (таб. 5.20).

Таблиця 5.20

Визначення меж встановлення ціни

№ п/п	*Рівень цін на товари-замінники*	*Рівень цін на товари-аналоги*	*Рівень доходів цільової групи споживачів*	*Верхня та нижня межі встановлення ціни на товар/послугу*
1	75000 грн	50000 грн	Стабільно високий, щоб мати можливість використовувати сервіс підтримки	Продаж продукту – 100$ за один екземпляр Підписка на оновлення – 50$ в місяць Додатковий пакет на support – 50$ в місяць

Наступним етапом є формування системи збуту (таб. 5.21).

Таблиця 5.21

Формування системи збуту

№ п/п	*Специфіка закупівельної и цільових клієнтів*	*Функції збуту, які має виконувати постачальник товару*	*Глибина каналу збуту*	*Оптимальна система збуту*
1	Клієнту для придбання ліцензії необхідно здійснити оформлення замовлення на сайті та здійснити електронну оплату через зручну для нього систему.	Розробити сайт з детальним описом програми. Забезпечити розробку зручної форми для оформлення замовлення та підключити безпечні швидкі способи оплати через розповсюджені системи. Надати можливість зворотнього зв'язку, підтримки клієнта.	Кількість посередників, що передають товар один одному до придбання клієнтом нульова	Збут через сайт через одноосібні та групові ліцензії на визначений термін.

Останнім етапом є розробка концепції маркетингових комунікацій (таб. 5.22).

Таблиця 5.22

Концепція маркетингових комунікацій

№ п/п	*Специфіка поведінки цільових клієнтів*	*Канали комунікацій, якими користуються цільові клієнти*	*Ключові позиції, обрані для позиціонування*	*Завдання рекламного повідомлення*	*Концепція рекламного звернення*
	Використання служби сапорту.	Telegram, slack, LinkedIn	Функціональність, інноваційність	Якомога найкраще донести до потенційних клієнтів, що вони потребують цей продукт	Продукт є зручним надійним та стабільним

Висновки до розділу 5

У п’ятому розділі визначена ідея стартапу, його сильні та слабкі сторони, здійснений технологічний аудит проекту, здійснене дослідження ринку, цільових груп клієнтів, їх характеристика, очікування та бажання. Проведений SWOT-аналіз, аналіз конкуренції, визначені стійкі конкурентоздатні властивості та переваги. Розроблена ринкова стратегія та маркетингова програма. Дослідження встановило, що продукт має унікальні характеристики, існують сегменти споживачів на ринку, які можуть бути зацікавлені в придбанні товару. Використовуючи стратегію диференціації як стратегію розвитку та стратегію розширення первинного попиту як базову стратегію конкурентної поведінки можна виходити на ринок.

ВИСНОВОК

Магістерську дисертаційну роботу присвячено розробці системи розпізнавання тексту на зображенні. Особливістю розробленої системи є здатність розпізнавання рукописних символів в умовах малої навчальної вибірки.

Наукова новизна полягає у розробці власного багатоетапного алгоритму розпізнавання тексту на зображенні, більш ефективного за аналоги. Розроблений алгоритм складається із етапу попередньої обробки зображення, що окрім стандартних елементів включає запропоноване удосконалення до відомих методів скелетизації зображення, та етапу класифікації, що включає запропонований метод формування структурної моделі символу та порівняння структурних моделей символів із еталонними за допомогою запропонованого критерію схожесті структурних моделей. Проведена перевірка ефективності розробленої системи показує високу ефективність запропонованого алгоритму у порівнянні із аналогами, здатними працювати при малій еталонній вибірці.

За підсумками виконання роботи отримані наступні наукові і практичні результати:

Досліджено існуючі системи і методи розпізнавання друкованих і рукописних символів.

Запропоновано покращений алгоритм скелетизації бінаризованого зображення.

Запропоновано структурну модель символу для застосування в рішенні задачі розпізнавання рукописних символів в умовах малої навчальної вибірки.

Розроблено алгоритм побудови запропонованої структурної моделі символу по растровому поданню його накреслення.

Вибрано критерій схожості структурних моделей символів.

Розроблено структурну схему програми, блок-схему роботи програми, діаграма класів і компонентів, інтерфейс програми

Реалізовано систему розпізнавання рукописних символів в умовах малої навчальної вибірки на основі застосування запропонованої структурної моделі символу і обраного критерію схожості структурних моделей символів.

Проведені обчислювальні експерименти з метою оцінки якості та ефективності розробленої системи.

Розроблена система розпізнавання тексту на зображенні має самостійне значення і, крім завдання розпізнавання символів, може застосовуватися для вирішення задач класифікації відбитків пальців, ідентифікації почерку, перевірки підписів на справжність та інших завдань, пов'язаних з аналізом бінарних зображень.

ПЕРЕЛІК ПОСИЛАНЬ

1. Фомин Я. А. Распознавание образов: теория и практика. Издание третье, дополненное / Я. А. Фомин // М.: ФАЗИС. – 2014. – 460 с.
2. Anderson N. Optical Character Recognition – Part 1 / N. Anderson // IMPACT Best Practice Guide. – 2010.
3. ABBYY FineReader [Электронный ресурс]. – Режим доступа: http://www.abbyy.ru/finereader/, свободный. – Загл. с экрана (дата обращения: 01.04.2017).
4. Cuneiform Windows [Электронный ресурс]. – Режим доступа: http://cognitiveforms.com/ru/products_and_services/ cuneiform, свободный. – Загл. с экрана (дата обращения: 01.04.2017).
5. Dynamsoft OCR SDK [Электронный ресурс]. – Режим доступа: http://www.dynamsoft.com, свободный. – Загл. с экрана (дата обращения: 01.04.2017).
6. Lam S. W. An Adaptive Approach to Document Classification and Understanding / S. W. Lam // IAPR Workshop on Document Analysis Systems, Kaiserslautern, Germany. – 1994. – P. 231-251.
7. Schantz H. F. History of OCR, Optical Character Recognition / Schantz H. F. // Recognition Technologies Users Association, 1982. – 114 p.
8. LeCun Y. Handwritten Zipcode Recognition With Multilayer Networks / Y. LeCun, O. Matan, B. Boser, J. S. Denker, D. Henderson,
9. R. E. Howard, W. Hubbard, L. D. Jackel, H. S. Baird // Proc. of International Conference on Pattern Recognition, Atlantic City. – 1990. – P. 541-551.
10. Lee S.-W. Off-line Recognition of Totally Unconstrained Handwritten Numerals Using Multilayer Cluster Neural Network / S.-W. Lee, Y. J. Kim // Proc. of the 12th IAPR International Conference on Pattern Recognition. Jerusalem, Israel. – 1994. – P. 507-509.
11. Krzyzak A. Unconstrained Handwritten Character Classification Using Modified Backpropagation Model / A. Krzyzak, W. Dai, C. Y. Suen // Proc. 1st Int. Workshop on Frontiers in Handwriting Recognition, Montreal, Canada. – 1990. – P. 155-166.
12. LeCun Y. Efficient BackProp / Y. LeCun, L. Bottou, G. B. Orr, K.-

13. Muller // Neural Networks: tricks of the trade. Springer. – 1998. – P. 9-48.
14. Ping Z. Documents filters using morphological and geometrical features of characters / Z. Ping, C. Lihui // Image and Vision Computing. 2001. – vol. 19. – P. 847-855.
15. Chen X. R. Detecting and Reading Text in Natural Scenes /
16. X. R. Chen // Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition – 2004. – P. 366-373
17. Knerr S. Handwritten digit recognition by neural networks with single-layer training / S. Knerr, L. Personnaz, G. Dreyfus // IEEE Transactions on Neural Networks 3 – 1992. – P. 962-968.
18. Lee Y. Handwritten digit recognition using K neares-neighbor, radialbasis function, and back-propagation neural networks / Y. Lee // Neural Computation 3, – 1991. – P. 440-449.
19. Martin G. L. Recognizing hand-printed letters and digits using backpropagation learning. / G. L. Martin, J. A. Pitman // Neural Computation 3. – 1991. – P. 258-267.
20. Seong-Wang L. Off-line Recognition of Totally Unconstrained Handwritten Numerals Using Multilayer Cluster Neural Network / L. Seong- Wang, J. K. Young // Proc. Of the 12th IAPR International Conference on Pattern Recognition. Jerusalem, Israel. – 1994. – P. 507-509.
21. Kan C. Invariant Character Recognition with Zernike and Orthogonal Fourier-Mellin Moments / C. Kan, M. D. Srinath // Pattern Recognition. – 2000. – Vol. 35. – P. 143–154.
22. Krizhevsky A. ImageNet Classification with Deep Convolutional Neural Networks / A. Krizhevsky, I. Sutskever, G. Hinton // Conference on Neural Information Processing Systems (NIPS). – 2012. – 27 p.
23. Vapnik V. N. Support Vector Networks / V. Vapnik, C. Cortes // Machine Learning. – 1995. – № 20(3). – P. 273-297.
24. Круглов В. В., Дли М. И., Голунов Р. Ю. Нечеткая логика и искусственные нейронные сети – М.: Физматлит, 2001. – 224 с.
25. Вапник В.Н., Червоненкис А.Я. Теория распознавания образов. М.: Наука, 1974. – 415 с.

26. Vapnik V. N. A Training Algorithm for Optimal Margin Classifiers /
27. V. N. Vapnik, E. Boser, I. M. Guyon // Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory. – 1992. – № 18. – P. 144-152.
28. Пытьев Ю. П. Морфологический анализ изображений / Ю. П. Пытьев // Докл. АН СССР. – Т. 269. – № 5. – 1983. – С. 1061-1064.
29. Пытьев Ю. П. Задачи морфологического анализа изображений / Ю. П. Пытьев // Математические методы исследования природных ресурсов Земли из космоса. – М: Наука, 1984. – С. 41-83.
30. Загоруйко Н. Г. Методы распознавания и их применение. – М.: Сов.радио, 1972. – 208 с.
31. Загоруйко Н. Г. Прикладные методы анализа данных и знаний. – Новосибирск: ИМ СО РАН, 1999. – 270 с.
32. Журавлев Ю. И. Распознавание. Математические методы. Программная система. Практические применения / Ю.И. Журавлев, В.В. Рязанов, О.В. Сенько. М.: ФАЗИС, 2006. – 176 с.
33. Журавлев Ю. И. Об алгебраическом подходе к решению задач распознавания или классификации / Ю. И. Журавлев // Проблемы кибернетики. – 1978. – Т. 33. – С. 5-68.
34. Коробейников А. П. Методы распознавания образов: Учебное пособие / А. П. Коробейников. – Ростов-на-Дону: Издательский центр ДГГУ, 1999. – 51 с.
35. Щепин Е. В. К топологическому подходу в анализе изображений / Е. В. Щепин, Г. М. Непомнящий // Межвузовский сборник научных трудов
36. «Геометрия, топология и приложения». – Москва, Мин. высшего и средн. спец. образ. РСФСР, Московский институт приборостроения. – 1990. – С. 13-25.
37. Спицын В. Г. Применение искусственных нейронных сетей для обработки информации / В.Г. Спицын, Ю.Р. Цой. – Томск: ТПУ, 2007. – 32 с.
38. Спицын В. Г. Применение генетического алгоритма для решения задач оптимизации / В. Г. Спицын, Ю. Р. Цой. – Томск: ТПУ, 2007. – 27 с.
39. Местецкий Л. М. Математические методы распознавания образов
40. / Л. М. Местецкий. – М.: МГУ, ВМиК, 2002. – 85 с.

41. Садыков С. С. Скелетизация бинарных изображений / С. С. Садыков, И. Р. Самандаров // Зарубежная радиоэлектроника. – 1985. – № 11. – С. 30-37.
42. Арлазаров В.Л., Астахов А.Д., Троянкер В.В., Котович Н.В. Адаптивное распознавание символов // Интеллектуальные технологии ввода и обработки информации. – 1998. – С. 39-56.
43. Арлазаров В.Л., Славин О.А. Алгоритмы распознавания и технологии ввода текстов в ЭВМ. // Информационные технологии и вычислительные системы. – 1996. – № 1. – С. 48-54.
44. Фаворская М. Н. Модель распознавания изображений рукописного текста / М. Н. Фаворская, А. Н. Горошкин // Вестник Сибирского государственного аэрокосмического университета имени академика М. Ф. Решетнева. – 2008. – № 2 (19).– С. 52-58.
45. Фаворская М. Н. Морфологическая обработка контурных изображений в системах распознавания текстовых символов / М. Н. Фаворская, А. С. Зотин, А. Н. Горошкин // Вестник Сибирского государственного аэрокосмического университета имени академика М. Ф. Решетнева. – 2007. – № 1 (14). – С. 70-75.
46. Хайкин С. Нейронные сети. Полный курс 2-е изд. Пер. с англ. – М.: Издательский дом «Вильямс». – 2006. – 1104 с.
47. Богданов В. Системы распознавания текстов в офисе / В. Богданов, К. Ахметов // Компьютер-пресс. – 1999. – № 3.– С. 40-42.
48. Аникеев М. В. Алгоритм распознавания бланков факсимильных сообщений. / М. В. Аникеев, В. М. Федоров, Н. Н. Цопкало // Известия ТРТУ. Специальный выпуск «Материалы XLVII научно-технической конференции»2002. – № 1(24). – С. 146-147.

ДОДАТКИ

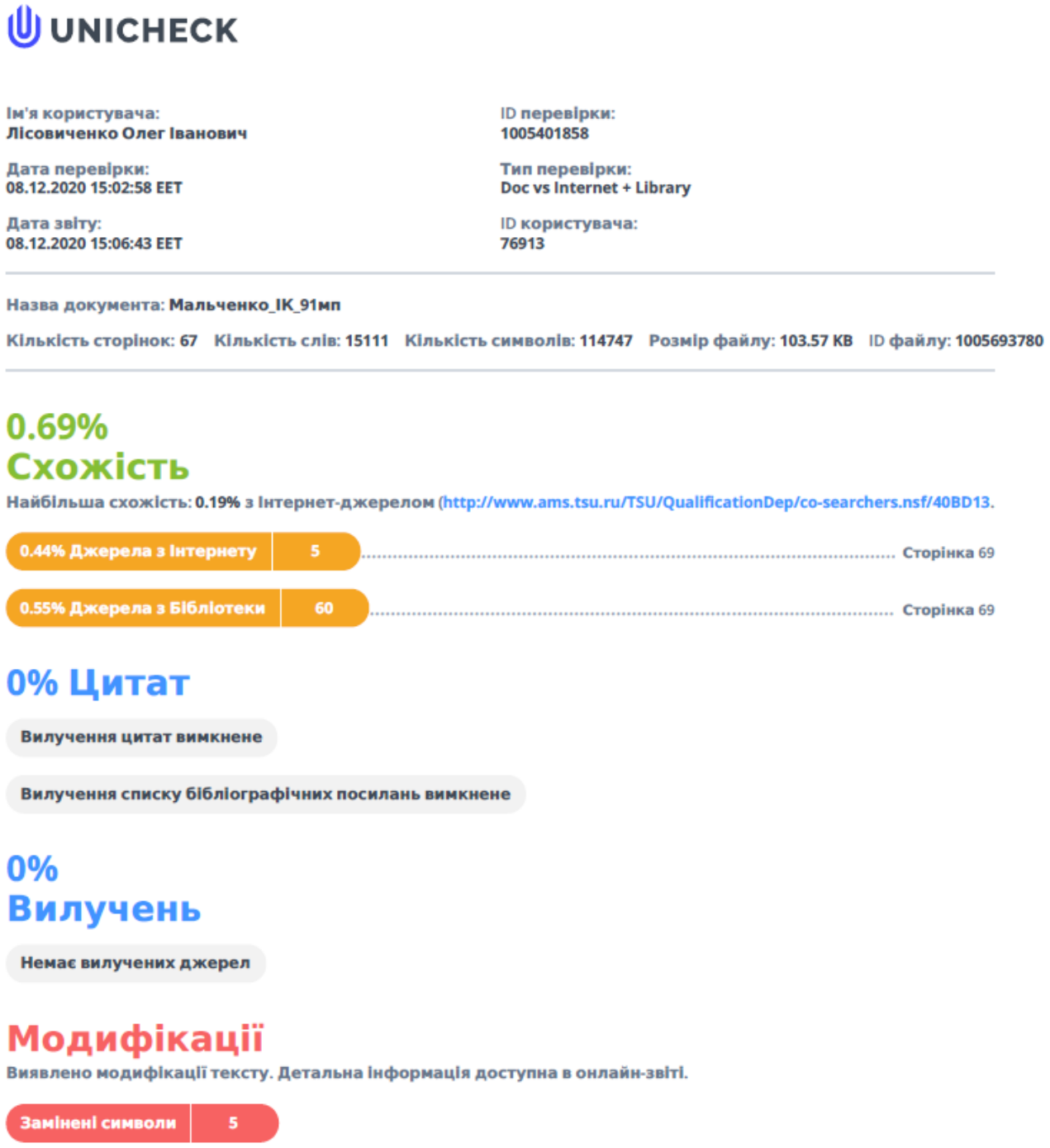
ДОДАТОК А

Відомість дипломного проекту

з/п	Формат	Позначення	Найменування	Кількість листів	Примітка
	А4		Завдання на дипломний проект		
	А4		Пояснювальна записка		
	А4		Додаток А. Відомість проекту		
	А4		Додаток Б. Результат перевірки на співпадіння		
	А3		Додаток В. Плакати		

ДОДАТОК Б

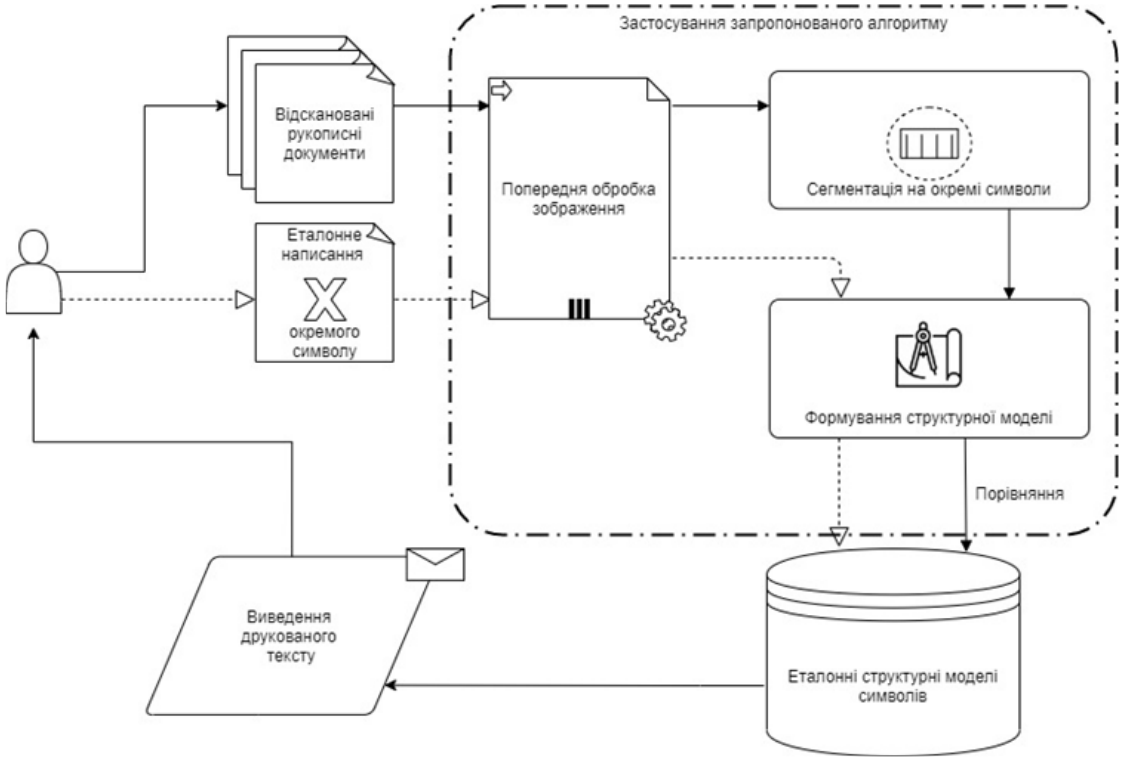
Результат перевірки на співпадіння



ДОДАТОК В

Плакати

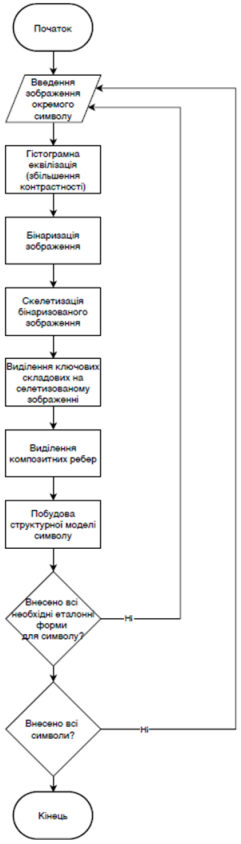
Загальна структурна схема розробленої системи



Демонстраційний плакат № 1
до дипломної роботи на тему
„ Розробка системи розпізнавання тексту на зображенні ”

Розробив: студент групи ІК-91мп Мальченко Є.Є.
Керівник: Ст. викладач Польшакова О.М.

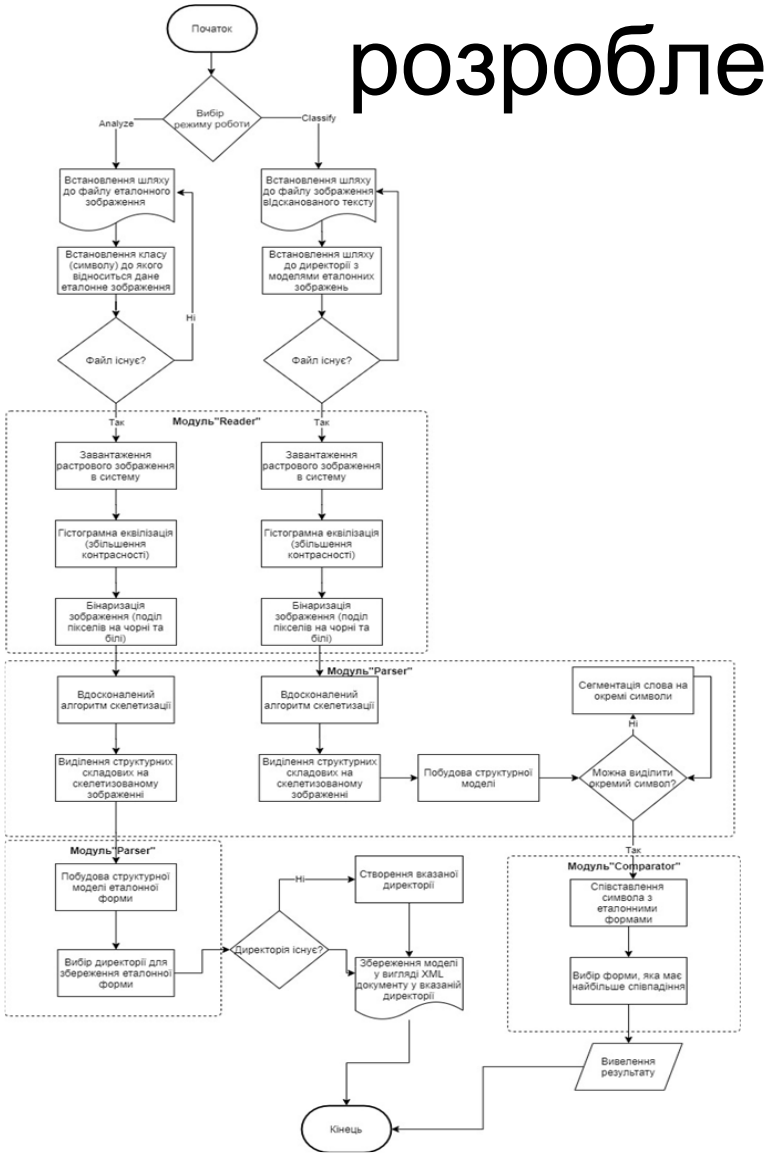
Загальна блок-схема розробленого алгоритму розпізнавання тексту на зображенні



Демонстраційний плакат № 2
до дипломної роботи на тему
„ Розробка системи розпізнавання тексту на зображенні ”

Розробив: студент групи ІК-91мп Мальченко Є.Є.
Керівник: Ст. викладач Польшакова О.М.

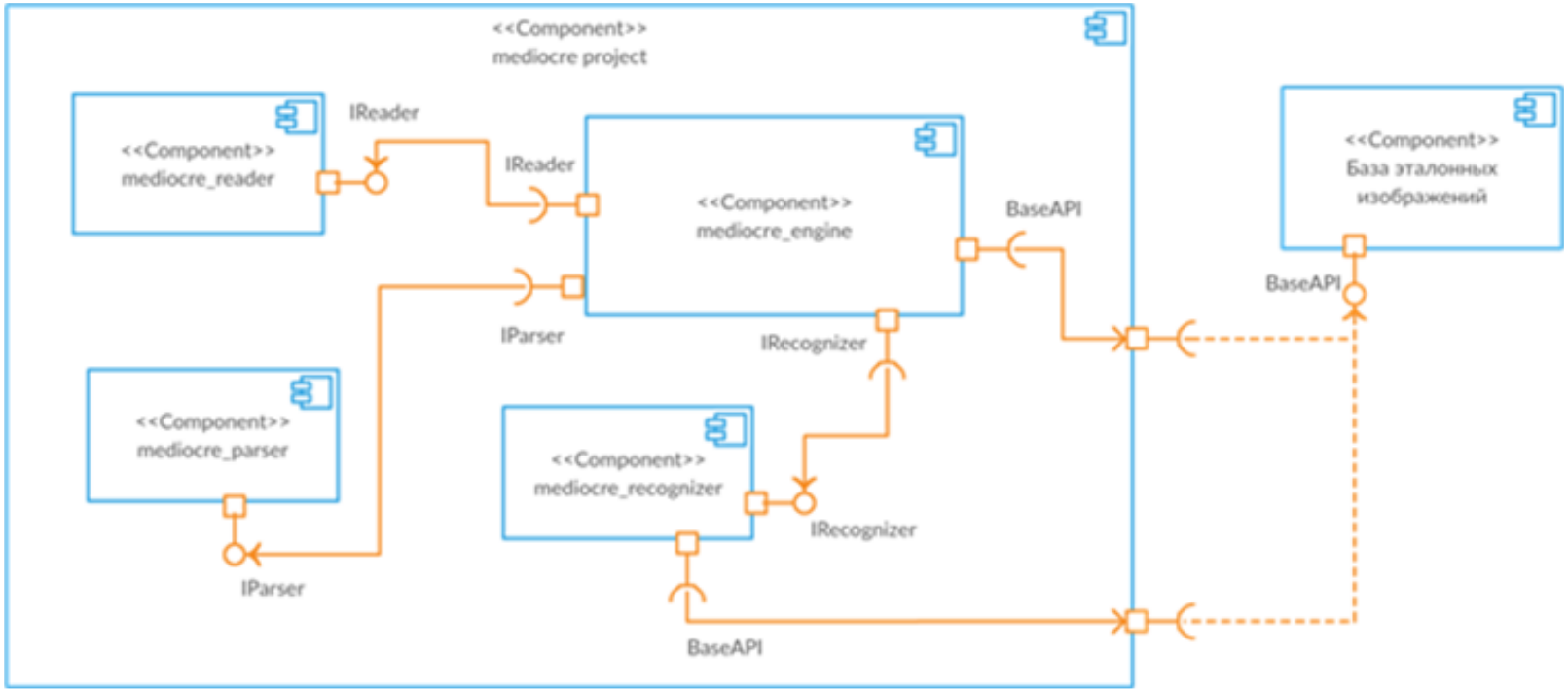
Загальна блок-схема роботи розробленої програми



Демонстраційний плакат № 3
до дипломної роботи на тему
„ Розробка системи розпізнавання тексту на зображенні ”

Розробив: студент групи ІК-91мп Мальченко Є.Є.
Керівник: Ст. викладач Польшакова О.М.

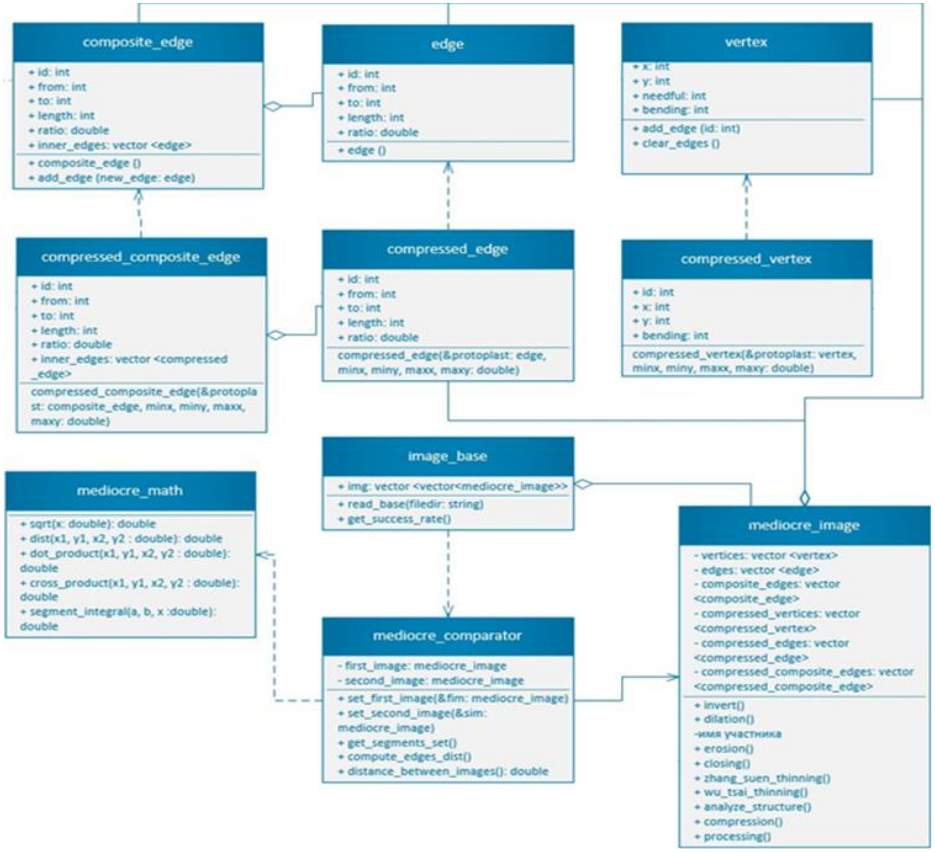
Діаграма компонентів розробленої системи



Демонстраційний плакат № 4
до дипломної роботи на тему
„ Розробка системи розпізнавання тексту на зображенні ”

Розробив: студент групи ІК-91мп Мальченко Є.Є.
Керівник: Ст. викладач Польшакова О.М.

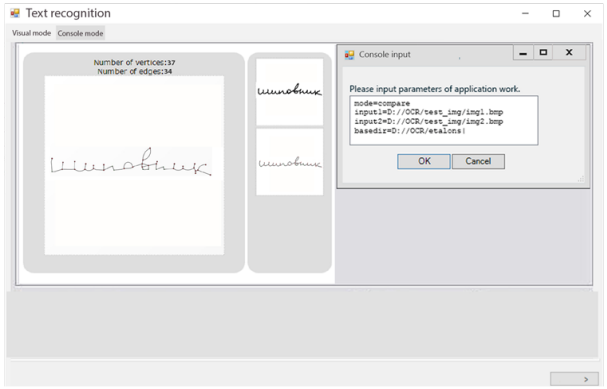
UML-Діаграма класів модуля розпізнавання символів

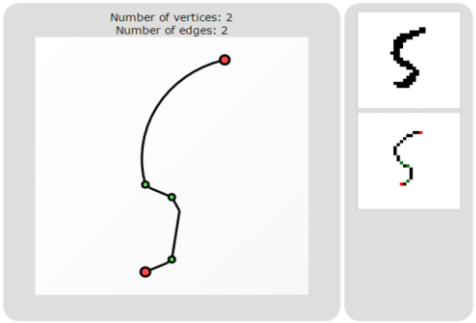


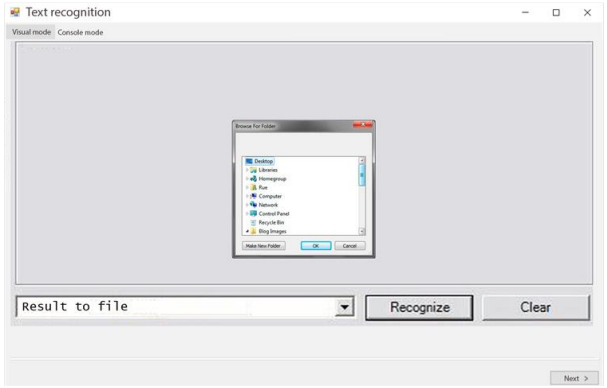
Демонстраційний плакат № 5
до дипломної роботи на тему
„ Розробка системи розпізнавання тексту на зображенні ”

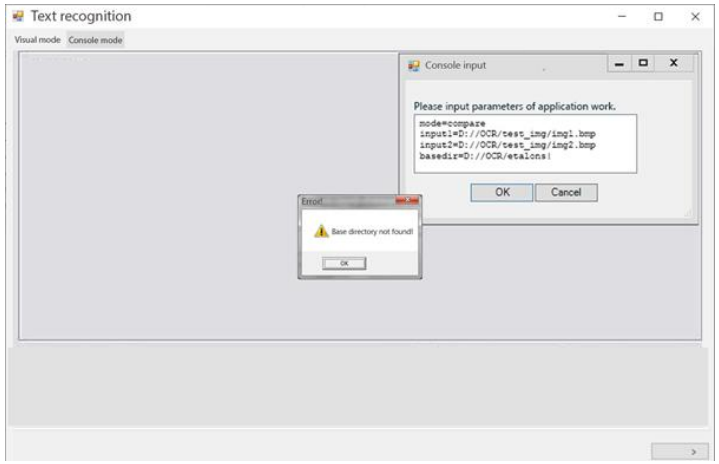
Розробив: студент групи ІК-91мп Мальченко Є.Є.
Керівник: Ст. викладач Польшакова О.М.

Приклади роботи розробленої програми









Демонстраційний плакат № 6
до дипломної роботи на тему
„ Розробка системи розпізнавання тексту на зображенні ”

Розробив: студент групи ІК-91мп Мальченко Є.Є.
Керівник: Ст. викладач Польшакова О.М.